Javier López

Optimización Multi-objetivo. Aplicaciones a problemas del mundo real.

Tesis Doctoral en Ciencias Informáticas
PREMIO DR. RAÚL GALLARD | Año 2014

edulp
Editorial de la Universidad de La Plata

Optimización Multi-objetivo.
Aplicaciones a problemas del mundo real.

Mg. Javier López

UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA

TESIS DOCTORAL EN CIENCIAS INFORMÁTICAS

Optimización Multi-objetivo. Aplicaciones a problemas del mundo real.

Mg. Javier López

Directores

Prof. Lic. Laura C. Lanzarini
Prof. Dr. Guillermo Leguizamón

La Plata, mayo de 2013

Editorial de la Universidad de La Plata

López, Javier
Optimización multi-objetivo: aplicaciones a problemas del mundo real. - 1a ed. - La Plata: EDULP, 2015.
308 p.; 24x16 cm.

ISBN 978-987-1985-59-3

1. Informática. 2. Educación Universitaria. I. Título
CDD 004.0711

Optimización Multi-objetivo
Aplicaciones a problemas del mundo real

Javier López

Diagramación: Andrea López Osornio

Editorial de la Universidad Nacional de La Plata (Edulp)
47 N.º 380 / La Plata B1900AJP / Buenos Aires, Argentina
+54 221 427 3992 / 427 4898
edulp.editorial@gmail.com
www.editorial.unlp.edu.ar

Edulp integra la Red de Editoriales de las Universidades Nacionales (REUN)

Primera edición, 2015
ISBN N.º 978-987-1985-59-3

Queda hecho el depósito que marca la Ley 11723
© 2015 - Edulp
Impreso en Argentina

Absurdo suponer que el paraíso
es sólo la igualdad, las buenas leyes
el sueño se hace a mano y sin permiso
arando el porvenir con viejos bueyes

Índice general

Agradecimientos

A Juan Scrinzi, con quién implementamos el primer algoritmo probabilistico, mas de 15 años atrás.

A Armando Di Giusti, por su inmediato apoyo cuando lo contacte con mi inquietud de hacer un doctorado en este tema.

A Laura Lanzarini, quien me guió, enseño, exigió y me acompaño durante todo el proceso. Sin su experiencia, trabajo y colaboración, no hubiera sido posible esta investigación.

A Guillermo Leguizamón, por su orientación y contribución en este extenso mundo de metaheurísticas evolutivas, y por todas las sugerencias y correcciones que mejoraron esta tesis en forma significativa.

A referentes de la comunidad científica por su apertura y predisposición para compartir información y discutir ideas. Tuve la oportunidad de tener relación directa como profesores y/o compañeros de congreso a Enrique Alba, Carlos Coello Coello, Fred Glover, Eckart Zitzler, Guillermo Luque, Gabriela Ochoa, Guillermo Simari, Ramón Garcia Martinez, Marcelo Fallapa, Remo Suppi, Fernando Tinneti. Seguro la lista es incompleta.

Agradezco a la empresa Ayuda Médica por permitirme llevar adelante este proyecto, especialmente a uno de sus directores, Sebastián Calderón, por su apoyo y confianza.

A toda la Universidad Nacional de La Plata, tanto al personal docente como no docente.

A mi familia, por el continuo apoyo y la paciencia.

Motivación

Resolver problemas es una motivación importante. Que esos problemas se encuentren entre los problemas más complejos que existen, es un impulso adicional para enfrentarlos.

La optimización de problemas es un terreno fértil en un mundo que se caracteriza por contar con recursos escasos (naturales, económicos, tecnológicos, infraestructura, sociales, tiempo, etc.). Hacer el mejor uso posible de estos recursos en una tarea, a la vez, importante y difícil. Ofrecer soluciones de calidad, aunque no necesariamente sean las mejores, implica que los recursos excedentes, frutos de la optimización, puedan utilizarse en nuevos productos o servicios.

La gran mayoría del software que utilizan las empresas tiene como misión principal la automatización de tareas repetitivas. Una minoría de aplicaciones de software se utiliza como soporte a la toma de decisiones de un decisor humano. Una porción ínfima de artefactos de software son capaces de ofrecer cual es la decisión adecuada para un problema complejo.

Es en este último grupo donde se encuentran las técnicas estudiadas en esta tesis. La implementación en el mundo real de algoritmos de búsqueda y optimización se hace necesaria y evidente a medida que aumenta la complejidad de los procesos, las empresas y gobiernos sufren una presión constante para ser más competitivos y eficientes, y los recursos disponibles se presentan como escasos ante una demanda en permanente aumento. Las metaheurísticas están pensadas para ofrecer una solución a este tipo de problemas pertenecientes a la clase de complejidad NP. Si bien son soluciones aproximadas, no exactas, en general son lo suficientemente buenas como para que su utilidad sea valiosa.

Objetivos

A continuación, se enumeran los objetivos del presente trabajo.

1. Estudiar el estado de avance actual de las metaheurísticas basadas en Inteligencia de Enjambres.
2. Evaluar las ventajas de la paralelización de las mismas.
3. Proponer mejoras incrementales a los algoritmos del tipo Optimización por Cúmulo de Partículas (PSO - *Particle Swarm Optimization*), en sus versiones para espacios continuos y para espacios binarios.
4. Estudiar los conceptos centrales de la optimización multi-objetivo.
5. Proponer mejoras a los algoritmos del tipo MOPSO (*Multi - Objective Particle Swarm Optimization*), comparando los resultados con metaheurísticas representativas del estado del arte.
6. Diseñar, desarrollar e implementar una herramienta de software para optimizar la atención domiciliaria de una empresa de emergencias médicas, basada en metaheurísticas evolutivas multi-objetivo.

Contribuciones

Entre las contribuciones del presente trabajo podemos destacar:

1. Relevamiento general de las técnicas de optimización.
2. Análisis de la optimización global (mono-objetivo) y con más de un objetivo (multi-objetivo), incluyendo para cada una aplicaciones y medidas de desempeño.
3. Implementación de una versión paralela del algoritmo PSO, en una red de computadoras heterogéneas y de bajo costo.
4. Desarrollo de una nueva versión del algoritmo PSO para espacios continuos con el agregado de un procedimiento de búsqueda local, mejorando los resultados reportados por la versión original.
5. Propuesta de mejora sobre el procedimiento de actualización de velocidad sobre el algoritmo PSO aplicado a espacio binarios.
6. Propuesta de una nueva versión de un algoritmo Optimización por Cúmulo de Partículas aplicado a problemas con más de un objetivo, agregando la posibilidad de variación del tamaño de la población de acuerdo a la complejidad del problema. Comparación de esta versión con las versiones de MOPSO con mejores resultados reportados en la literatura, y con otras metaheurísticas referentes en optimización multi-objetivo.
7. Puesta en marcha de un proceso de optimización automático, en una empresa del mundo real de la industria de la salud, logrando mejoras sustanciales en las actividades afectadas.

Resumen

Cuando hablamos de optimización en el ámbito de las ciencias de la computación hacemos referencia al mismo concepto coloquial asociado a esa palabra, la concreción de un objetivo utilizando la menor cantidad de recursos disponibles, o en una visión similar, la obtención del mejor objetivo posible utilizando todos los recursos con lo que se cuenta.

Los métodos para encontrar la mejor solución (óptima) varían de acuerdo a la complejidad del problema enfrentado. Para problemas triviales, el cerebro humano posee la capacidad de resolverlos (encontrar la mejor solución) directamente, pero a medida que aumenta la complejidad del problema, se hace necesario contar con herramientas adicionales.

En esta dirección, existe una amplia variedad de técnicas para resolver problemas complejos. Dentro de estas técnicas, podemos mencionar las técnicas exactas. Este tipo de algoritmos son capaces de encontrar las soluciones óptimas a un problema dado en una cantidad finita de tiempo. Como contrapartida, requiere que el problema a resolver cumpla con condiciones bastante restrictivas.

Existen además un conjunto muy amplio de técnica aproximadas, conocidas como metaheurísticas. Estas técnicas se caracterizan por integrar de diversas maneras procedimientos de mejora local y estrategias de alto nivel para crear un proceso capaz de escapar de óptimos locales y realizar una búsqueda robusta en el espacio de búsqueda del problema. En su evolución, estos métodos han incorporado diferentes estrategias para evitar la convergencia a óptimos locales, especialmente en espacios de búsqueda complejos. Este tipo de procedimientos tienen como principal característica que son aplicables a cualquier tipo de problemas, sin requerir ninguna condición particular a cumplir por los mismos. Estas técnicas no garantizan en ningún caso la obtención de los valores óptimos de los problemas en cuestión, pero se ha demostrado que son capaces de alcanzar muy buenos valores de soluciones en períodos de tiempo cortos. Además, es posible aplicarlas a problemas de diferentes tipos sin mayores modificaciones, mostrando su robustez y su amplio espectro de uso.

La mayoría de estas técnicas están inspiradas en procesos biológicos y/o físicos, y tratan de simular el comportamiento propio de estos procesos que favorecen la búsqueda y detección de soluciones mejores en forma iterativa. La más difundida de estas técnicas son los algoritmos

genéticos, basados en el mecanismo de evolución natural de las especies.

Existen diferentes tipos de problemas, y multitud de taxonomías para clasificar los mismos. En el alcance de este trabajo nos interesa diferenciar los problemas en cuanto a la cantidad de objetivos a optimizar. Con esta consideración en mente, surge una primera clasificación evidente, los problemas mono-objetivo, donde existe solo una función objetivo a optimizar, y los problemas multi-objetivo donde existe más de una función objetivo. En el presente trabajo se estudia la utilización de metaheurísticas evolutivas para la resolución de problemas complejos, con uno y con más de un objetivo. Se efectúa un análisis del estado de situación en la materia, y se proponen nuevas variantes de algoritmos existentes, validando que las mismas mejoran resultados reportados en la literatura.

En una primera instancia, se propone una mejora a la versión canónica y mono-objetivo del algoritmo PSO, luego de un estudio detallado del patrón de movimientos de las partículas en el espacio de soluciones. Estas mejoras se proponen en las versiones de PSO para espacios continuos y para espacios binarios. Asimismo, se analiza la implementación de una versión paralela de esta técnica evolutiva.

Como segunda contribución, se plantea una nueva versión de un algoritmo PSO multiobjetivo (*MOPSO Multi Objective Particle Swarm Optimization*) incorporando la posibilidad de variar dinámicamente el tamaño de la población, lo que constituye una contribución innovadora en problemas con mas de una función objetivo.

Por último, se utilizan las técnicas representativas del estado del arte en optimización multi-objetivo aplicando estos métodos a la problemática de una empresa de emergencias médicas y atención de consultas domiciliarias. Se logró poner en marcha un proceso de asignación de móviles a prestaciones médicas basado en metaheurísticas, logrando optimizar el proceso de asignación de móviles médicos a prestaciones médicas en la principal compañía de esta industria a nivel nacional.

Publicaciones derivadas

Publicaciones en Congresos Internacionales

- **Particle Swarm Optimization with Oscillation Control**. Lopez Javier, Lanzarini Laura, De Giusti Armando. Genetic and Evolutionary Computation Conference. ACM GECCO Proceeding. ISBN:978-1-60558-325-9. Montréal, Canada. July 2009.
- **VarMOPSO: Multi-Objective Particle Swarm Optimization with Variable Population Size**. Lopez Javier, Lanzarini Laura, De Giusti Armando, Advances in Artificial Intelligence. IBERAMIA 2010, Lecture Notes in Computer Science, Springer Berlin / Heidelberg.
- **A New Binary PSO with Velocity Control**, Lanzarini Laura, López Javier, Maulini Juan,De Giusti Armando, Advances in Swarm Intelligence - Second International Conference, ICSI 2011, Chongqing, China, June 12-15, 2011, Proceedings, Part I.
- **Evolutionary Multiobjective Optimization for Emergency Medical Services**, Lopez Javier, Lanzarini Laura, De Giusti Armando, 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, Companion Material Proceedings, Dublin, Ireland, July 12-16, 2011.
- **Variable Population MOPSO Applied to Medical Visits**, Lopez Javier, Lanzarini Laura, Aurelio Fernandez Bariviera, Fuzzy Economic Review, Vol XVII, No. 1., September 2012.

Publicaciones en Workshops de la Especialidad

- **Metaheurísticas Poblacionales Aplicadas a la Resolución de Problemas Complejos**. Lanzarini Laura, López Javier. XI Workshop de Investigadores en Ciencias de la Computación (WICC 2009), Área Agentes y Sistemas Inteligentes.Mayo de 2009. San Juan.
- **Minería de Datos Utilizando Estrategias adaptativas. Aplicaciones en Optimización de Procesos y Modelización**.

Lanzarini Laura, López Javier, XII Workshop de Investigadores en Ciencias de la Computación (WICC 2010). Area Agentes y Sistemas Inteligentes. Mayo de 2010. Santa Cruz. Argentina.

- **Sistemas Inteligentes. Aplicaciones en Optimización de Procesos, Minería de Datos, Procesamiento de señales y Robótica Evolutiva**, Lanzarini Laura, Hasperue Waldo, Corbalán Leonardo, López Javier, Estrebou César, Ronchetti Franco, Maulini Juan, Villa Monte Augusto, XIII Workshop de Investigadores en Ciencias de la Computación (WICC 2011). Area Agentes y Sistemas Inteligentes. Mayo de 2011. Rosario. Argentina.

- **Técnicas de Optimización. Aplicaciones en Minería de Datos, y Robótica Evolutiva** Lanzarini Laura, Hasperue Waldo, Corbalán Leonardo, López Javier, Estrebou César, Ronchetti Franco, Maulini Juan, Villa Monte Augusto, XIII Workshop de Investigadores en Ciencias de la Computación (WICC 2012). Area Agentes y Sistemas Inteligentes.

Estructura de Documento

La presente investigación se estructura en tres partes principales:

1. Aspectos Teóricos
2. Optimización por Cúmulo de Partículas. Nuevas Variantes
3. Caso de Estudio

La primera de estas partes está dividida en tres capítulos. En el primero de ellos, la investigación se centra en el estudio, definición y análisis del concepto de optimización, diferenciando entre la optimización local y la optimización global. En esta parte del documento de describen los conceptos básicos relacionados con los procedimientos de búsqueda y optimización, aplicables a cualquier técnica creada para este fin.

En el capítulo 2 se efectúa una categorización de las técnicas de optimización existentes, introduciendo y definiendo el concepto de metaheuristicas. Luego, se hace referencia a la computación evolutiva en general, y se describe en detalle los componentes de los Algoritmos Genéticos (AGs), la primera y más utilizada de las técnicas en este campo de estudio. Luego se presentan las principales dificultades que presentan los problemas de optimización en general.

En el capítulo 3 perteneciente a esta primera parte, se detallan los tipos de optimización existentes en función de la cantidad de funciones objetivos a optimizar. En este capítulo se analiza, además, los procedimientos utilizados para efectuar medición de performance de algoritmos de optimización estocásticos, mono y multi-objetivo. Para finalizar, se hace una revisión del teorema de *No Free Lunch*, de alta relevancia para este tópico, y se analiza el impacto del mismo en los problemas de optimización.

En la segunda parte de documento se profundiza el estudio en la metaheurísticas de Optimización por Cúmulo de Partículas (*PSO - Particle Swarm Optimization*), dentro del ámbito de la inteligencia de enjambres (*Swarm Intelligence*). Esta parte está compuesta por cinco capítulos.

En el capítulo 4, primer capítulo de esta segunda parte, se describe en detalle la metaheurística Optimización por Cúmulo de Partículas (PSO), haciendo referencia a sus antecedentes y destacando las mejoras principales incorporadas sobre la estructura del algoritmo base.

A continuación, en los siguientes 4 capítulos, se describen propuestas de algoritmos del tipo PSO aplicadas a espacios continuos y binarios, y a problemas mono-objetivo y multi-objetivo.

En el capítulo 5 se propone una variante paralela del algoritmo PSO, para ejecutar en redes de computadoras homogéneas, comparando el rendimiento con la versión canónica del algoritmo y realizando mediciones de *speed-up* alcanzado.

En el capítulo 6 se presenta una versión que mejora el proceso de aproximación final de las partículas a los puntos óptimos locales del espacio de soluciones. Este objetivo se logra con la incorporación de un procedimiento de búsqueda local del tipo determinístico, que se aplica solamente en aquellos casos donde las partículas oscilan alrededor de algún punto óptimo local (global).

El capítulo 7 expone una variante del algoritmo PSO aplicado a funciones con dominios binarios, presentándose una novedosa y eficiente ecuación de actualización de la velocidad y ubicación de las partículas.

En el capítulo 8 se expone una versión del algoritmo de Optimización por Cúmulo de Partículas aplicado, en esta ocasión, a problemas multi-objetivo. Se introduce el concepto de población de tamaño variable, una variante muy poca explorada en este tipo de problemas.

La tercera parte del documento se focaliza en la aplicación de metaheurísticas evolutivas a problemas del mundo real con más de un objetivo, y está dividido en tres capítulos.

El primero de ellos, capítulo 9, describe cómo es el proceso de negocio de una compañía de emergencias y prestaciones médicas domiciliarias. Se realiza un relevamiento detallado de los procesos de negocio directamente relacionados con el presente trabajo.

El capítulo 10 describe la solución propuesta, y detalla cómo la misma se integra con el resto de las herramientas de software de la empresa. Este capítulo es importante para entender el contexto de aplicación de la técnica de optimización elegida, así como para entender los trabajos previos requeridos para aplicar la misma en este caso del mundo real.

El capítulo 11 se enfoca en la solución algorítmica especifica propuesta, y en los experimentos y mediciones efectuados para respaldar la elección realizada.

Por último, se incluye un párrafo de conclusiones finales como cierre del presente trabajo de investigación, así como la propuesta de líneas de estudio futuras de acuerdo a lo aprendido.

PARTE I

Aspectos Teóricos

Métodos de Optimización - Conceptos Básicos

Todos los métodos de optimización, sean éstos de naturaleza estocástica o determinística, poseen componentes en común. Este capítulo de la presente tesis describe cada uno de estos componentes, con el objetivo de generar el marco de referencia que permita lograr un entendimiento acabado de los temas por venir. Los conceptos teóricos incluidos en el presente capítulo permiten abordar el problema de optimización desde la Teoría de Conjuntos. A lo largo de este trabajo, las referencias a esta teoría fundamental de la matemática es permanente e inevitable.

1.1 Introducción

Puede definirse a las técnicas de optimización como aquellos métodos que permiten seleccionar la mejor solución ante un conjunto de alternativas disponibles. Esta definición se ajusta a todos los campos donde se aplica la optimización, la matemática, las ciencias de la computación, la economía y los negocios.
Esta definición se relaciona con el concepto de eficiencia, que remite a lograr el mejor resultado posible dada la utilización de recursos efectuada.
Por lo tanto, se está optimizando cuando dada una cantidad de recursos definida, se obtiene el máximo posible del producto o servicio objetivo buscado, o, cuando dada una cantidad de producto o servicio objetivo buscado, se utiliza la menor cantidad de recursos para producir el mismo.
La utilización de herramientas provenientes del campo de la matemática se remonta a 1758, cuando economistas comenzaron a describir sistemas económicos en términos matemáticos [1], pero puede decirse que el campo de optimización logró sus primeros avances significativos entre los años 1939 y 1947, gracias a los progresos en la rama de programación lineal desarrollados por el matemático ruso Leonid Kantorovich, y por el trabajo de los siguientes matemáticos: el norteamericano George B. Dantzig, autor del método simplex y el matemático húngaro-norteamericano John von Neumann, quien

propuso la teoría de la dualidad. A partir de este punto, las técnicas de optimización se convirtieron en un campo de fuerte desarrollo en la matemática, economía y en las ciencias de la computación.

1.2 Definiciones

Podríamos decir que el concepto de función como un objeto matemático independiente sienta las bases para poder definir un problema de optimización. Este concepto fue evolucionando desde los orígenes del calculo en el siglo XVII, hasta llegar a una definición más elaborada y completa con el desarrollo de la Teoría de Conjuntos en los siglos XIX y XX. A continuación se incluyen las definiciones más relevantes relacionadas con problemas de optimización, considerando los conceptos matemáticos subyacentes.

Problemas

La optimización busca obtener los mejores elementos $\mathbf{x}^*$ a partir de un conjunto X según un criterio $f(x)$. Este criterio se expresa a través de una función matemática denominada función objetivo.

Matemáticamente, se trata de obtener los puntos extremos de una función. En el caso de un problema de minimización, dada una función $f: X \in R^n$, R se trata de hallar el vector $x \in X$ que minimice dicha función como se indica en (?)

$$\min_{x \in X} f(x) \qquad (1.1)$$

sujeto a

$$g_i(\mathbf{x}) \mathrm{d} 0 i = 1, \ldots, q \qquad (1.2)$$

$$h_j(\mathbf{x}) = 0 j = q+1, \ldots, m \qquad (1.3)$$

donde X es el dominio de la función o espacio de búsqueda, Y el codominio o espacio objetivo, y $f(x)$ la función para la cual se desea obtener el valor mínimo o función objetivo.

Las restricciones definidas acotan el espacio de soluciones, dividiendo el espacio de búsqueda en dos regiones:

- Soluciones Factibles: Aquellos elementos del espacio de búsqueda que cumplen con todas las ecuaciones de restricción.
- Soluciones No Factibles: Aquellos elementos del dominio de la función que no cumplen con al menos una de las restricciones del problema.

Los métodos de optimización más básicos son aquellos que se utilizan para resolver problemas de optimización no lineales de variables reales en un espacio continuo, sin restricciones. En estos problemas, se asume que $f(x)$ es dos veces continuamente diferenciable.

La mayoría de los métodos para optimizar este tipo de funciones son algoritmos iterativos basados en la expansión de las series de Taylor de $f(x)$, requiriendo el cálculo de derivadas parciales [2]. Muchas de las técnicas de optimización existentes estiman las derivadas parciales utilizando diferencias finitas. De esa forma, si la derivada no se puede resolver analíticamente, o si no se desea calcularla, estimando la misma por diferencias aún es posible resolver el problema.

Ascendiendo en orden de complejidad, se encuentran los problemas de programación lineal con restricciones. Estos problemas son de suma importancia dado que son ampliamente aplicables en problemas del mundo real en el campo de la planificación y la economía, existiendo mucho material teórico desarrollado en este campo.

La Programación Lineal es uno de los responsables principales del desarrollo del campo de la optimización. En este tipo de problemas la función objetivo $f(x)$ es lineal y está sujeta a restricciones de igualdad y desigualdad, también lineales [3].

Una sorprendente cantidad de investigación en este tópico se concentró entre 1938 y 1947, destacándose el desarrollo por parte de G.B. Dantzig del método simplex y la teoría de la dualidad de von Neumann [4] [5].

En este tipo de problemas está demostrado que la solución óptima siempre se corresponde con uno de los vértices del poliedro P formado por el conjunto de restricciones del problema. La idea del método *Simplex* es sumamente simple, consiste en encontrar uno de los vértices del poliedro. Luego, se mueve cuesta abajo respecto de los valores de $f(x)$ entre vértice y vértice a través de los bordes de P; de esta forma, se genera una secuencia de vértices con valores estrictamente decrecientes. En una serie finita de pasos, se encuentra el vértice con menor valor de $f(x)$ o un lado con extensión ilimitada. En este último caso, el problema no tiene solución.

El siguiente conjunto de problemas a mencionar son los problemas de optimización de funciones suaves no lineales con restricciones suaves,

sobre un conjunto finito de variables reales. Dentro de este conjunto de problemas, se encuentran los problemas cuadráticos, donde $f(x)$ es una función cuadrática y las restricciones son lineales. Los métodos desarrollados para resolver este tipo de problemas se basan en remplazar el problema original por un problema más fácil. De esta forma, estas técnicas conducen a la solución de una secuencia de subproblemas, relacionados de alguna forma con el problema original.

Algunas de las técnicas implican la minimización de problemas sin restricciones o el tratamiento de límites y restricciones lineales derivadas de las restricciones del problema original.

La mayoría de estos métodos usan diferentes clases de funciones de penalidad $p(x)$, transformando el problema original de una función no lineal $f(x)$ con restricciones a otro problema consistente en la minimización de una función $i(x)$ sin restricción, siendo $i(x)=f(x)+p(x)$.

Existen además propuestas basadas en generar una secuencia de puntos estrictamente dentro de la zona de soluciones factibles, requiriendo la existencia de un punto inicial en esta región del espacio de búsqueda. Estos métodos utilizan diferentes clases de funciones de barrera $b(x)$, transformando el problema original de una función no lineal $f(x)$ con restricciones a otro problema consistente en la minimización de una función $j(x)$ sin restricción, siendo $j(x)=f(x)+b(x)$. La función $b(x)$ incluye un término que asegura que la secuencia generada se mantiene dentro de la zona factible [6].

Existe un grupo de problemas particulares donde una o varias restricciones requieren que las variables del problema sean enteras. Este tipo de problemas se conocen como programación entera u optimización combinatoria, y se caracterizan por tener espacios de búsqueda (dominio de la función) discretos [7].

Considérese ahora el caso de funciones no diferenciables, donde $f(x)$ no es continua ni es posible calcular su derivada para todos los puntos. Este tipo de problemas son difíciles de resolver, dado que no existen soluciones analíticas vinculadas con los mismos.

Para la optimización de éste tipo de funciones complejas aparecen los algoritmos probabilísticos. Estos algoritmos realizan muestreos del espacio de búsqueda intentando descubrir patrones regulares en los mismos [8]. Estas técnicas se guían evaluando la función objetivo a optimizar en cada uno de los puntos visitados, intentando obtener información relevante del problema a los efectos de dirigir la exploración a las zonas más promisorias del espacio de búsqueda.

Función Objetivo

Una función objetivo es una función matemática sujeta a optimización (maximización o minimización).
El codominio de esta función es un subconjunto de los números reales.
El dominio puede representar cualquier tipo de elementos como ser números reales, enteros, binarios, listas, tareas a realizar, recursos disponibles, etc. Este dominio se conoce como espacio del problema. Las funciones objetivo van desde meras funciones matemáticas hasta complejos algoritmos de cálculo.
La función objetivo define el problema a optimizar, y devuelve como resultado un valor real o un vector de valores reales que representan la utilidad, respecto del problema en ciernes, de los valores correspondientes del dominio.

Espacio del Problema

El espacio del problema *X* es el conjunto formado por todos los elementos posibles del problema de optimización. Este espacio puede estar acotado por restricciones, que delimitan las soluciones viables.
Estas restricciones pueden ser lógicas o prácticas. Respecto del primer caso, puede haber reglas (restricciones) que definen que algún elemento no puede ser una solución. Respecto del segundo, hay límites en la capacidad de procesamiento, por ejemplo una computadora no puede representar los números reales con una precisión infinita. Este espacio puede ser finito o infinito. Ejemplos de espacios del problema son los números naturales, los reales, binarios o cualquier otra representación matemática.

Variables de Decisión

El espacio de un problema puede tener *n* dimensiones. Se define como variables de decisión a las variables *n-dimensionales* que identifican cada solución en el espacio de del problema *X* en forma univoca.

Espacio Objetivo

El espacio objetivo es el espacio abarcado por el co-dominio de la función objetivo. Las dimensiones del espacio objetivo se corresponden con la cantidad de funciones que se están optimizando.
Cuando se optimiza una única función objetivo, se denomina optimización mono-objetivo. Cuando existe más de una función objetivo, se denomina optimización multi-objetivo (Multi Objective

Optimization). Cuando los problemas tienen 4 o más funciones objetivos, la escalabilidad de los algoritmos de optimización multi-objetivo se ve afectada, por lo que se generó una nueva rama de investigación denominada Many Objective Optimization.

Restricciones

En algunos problemas existen restricciones de igualdad y restricciones de desigualdad con la que deben cumplir las soluciones para ser una solución válida. Ver ecuaciones 1.2 y 1.3.
Estas restricciones dividen el espacio de búsqueda en regiones factibles y regiones no factibles (figura 1.1). Usualmente, la proporción de las mismas, así como su distribución en el espacio del problema definen la complejidad de un problema de optimización.

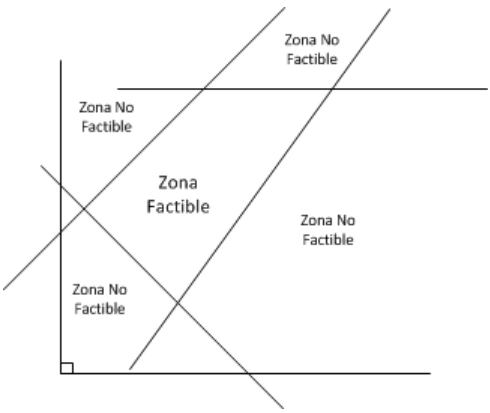


Figura 1.1: Gráfico de región factible y no factible

Espacio de Búsqueda

El espacio de búsqueda *G* es una representación (mapeo) del espacio del problema *X* y es donde los algoritmos de optimización efectúan las operaciones de búsquedas de nuevos elementos, potenciales soluciones al problema. Dependiendo del problema a optimizar, el espacio del problema y el espacio de búsqueda pueden ser iguales.

La función de mapeo entre el espacio del problema y el espacio de búsqueda se denomina representación del problema.

Se define como espacio de búsqueda *G* de un problema de optimización al conjunto de todos los elementos *g* que pueden ser procesados por procedimientos de búsquedas [9].

Se puede conceptualizar como la implementación del espacio del problema.

Solución Candidata

Una solución candidata es un elemento del espacio del problema *X* que puede ser una solución posible al problema de optimización. Estas soluciones se encuentran dentro de la región factible de espacio del problema, por lo que deben satisfacer todas las restricciones de igualdad y desigualdad asociadas con la función objetivo a optimizar.

Espacio de Soluciones

El espacio de soluciones *S* es el conjunto formado por la unión de todas las soluciones candidatas del problema.

$$\mathbf{x}^* \subseteq \mathbf{S} \subseteq \mathbf{X} \qquad (1.4)$$

Este conjunto forma el total de los puntos que conforman las regiones factibles del espacio del problema. La solución al problema de optimización, $\mathbf{x}^*$, debe encontrarse incluido dentro de este conjunto. Es de interés la exploración de las zonas fronterizas entre las regiones factibles y no factibles, dado que en muchos casos allí se encuentran los óptimos globales del problema [10].

Representación (Relación espacio del problema - espacio de búsqueda)

El mapeo entre el espacio del problema *X* y el espacio de búsqueda *G* es una relación binaria (gpm - ontogenic mapping) *gpm*:*G*' *X* del tipo left-total que vincula los elementos del espacio de búsqueda *G* con elementos del espacio del problema *X* [11].

$$\forall g \in G, \exists x \in X : gmp(g) = x \qquad (1.5)$$

Normalmente, *gpm* es una función determinística. La función de mapeo *gpm* debería ser sobreyectiva [12], para asegurar que todos los elementos del espacio del problema *X* estén relacionados con al menos un elemento

del espacio de búsqueda *G*. Obviamente, si hay elementos del codominio que no tienen imagen en el espacio de búsqueda (dominio), nunca podrían ser soluciones candidatas al problema de optimización.

Además, si la función *gpm* es biyectiva puede construirse un mapeo inverso $gmp^{-1}:X' \; G$; es decir que existe una relación 1 a 1 entre todos los elementos del dominio y codominio, entonces puede afirmarse que ese espacio de búsqueda está libre de redundancia.

$$gpm^{-1}(x)=q \Leftrightarrow gpm(g)=x \forall, x \in X, g \in G \qquad (1.6)$$

Esta función debería asegurar que los puntos óptimos (locales / globales) del espacio de búsqueda *G* se correspondan con los puntos óptimos del espacio del problema *X*. Esta última propiedad es central para que los puntos óptimos a evaluar por el algoritmo de optimización se correspondan con las soluciones óptimas del problema real.

Operaciones de Búsqueda

Las operaciones de búsqueda *searchOP* son las operaciones utilizadas para recorrer el espacio de búsqueda y encontrar la solución al problema de optimización. El resultado de una operación de búsqueda es un elemento del espacio de búsqueda.

$$searchOp:G^n \Longrightarrow G \qquad (1.7)$$

Una operación de búsqueda puede recibir como parámetros de entrada uno o más puntos del espacio de búsqueda visitado. Estas operaciones de búsqueda definen la secuencia de puntos a visitar por el algoritmo y por lo tanto su rol es central en la resolución del problema.

Las operaciones de búsqueda son en general probabilísticas, dado que normalmente poseen una componente estocástica.

Un conjunto *Op* de operaciones de búsqueda es **completo** si y solo si cada punto g_1 del espacio de búsqueda *G* puede ser alcanzado a partir de cualquier otro punto $g_2 \in G$ aplicando sólo operaciones de búsqueda $searchOP \in Op$

$$\forall g_1, g_2 \in G \Rightarrow \exists k \in N : P(g_1 = Op^k(g_2)) > 0 \qquad (1.8)$$

Nótese que la ecuación 1.8 indica la obtención de g_1 con una cierta probabilidad. Esto indica que las operaciones de búsqueda no son determinísticas.

Un conjunto de operaciones de búsqueda es **débilmente completo** si y solo si cada punto del espacio de búsqueda *G* puede ser alcanzado aplicando sólo operaciones de búsqueda *searchOP*∈*Op*. Por lo tanto, un conjunto de operaciones de búsqueda **débilmente completo** incluye al menos una función sin parámetros.

$$\forall g \in G \Rightarrow \exists k \in N : P(g = Op^k) > 0 \qquad (1.9)$$

Si el conjunto de operaciones de búsqueda es incompleto, existirán puntos del espacio de búsqueda que nunca podrán ser alcanzados. Esto no es una condición deseable para un algoritmo de optimización.

Adyacencia en el Espacio de Búsqueda

El punto g_2 es adyacente al punto g_1 en el espacio de búsqueda *G* si puede ser alcanzado aplicando una simple operación de búsqueda sobre g_1. Esta relación no requiere simetría.

$$adyacente(g_2, g_1) = \begin{cases} True\ si\ \exists searchOp \in Op : P(searchOp(g_1) = g_2) > 0 \\ False\ en\ caso\ contrario \end{cases} \qquad (1.10)$$

Adyacencia en el Espacio del Problema

El punto x_2 es adyacente del punto x_1 en el espacio del problema *X* si puede ser alcanzado aplicando una única operación de búsqueda *searchOp* sobre sus correspondientes elementos en el espacio de búsqueda.

$$adyacente(x_2, x_1) = \begin{cases} True\ si\ \exists g_1, g_2 : x_1 = gpm(g_1) \wedge x_2 = gpm(g_2) \wedge adyacente(g_2, g_1) \\ False\ en\ otro\ caso \end{cases} \qquad (1.11)$$

Gradiente

El gradiente de un campo escalar $f:R^n, R$ es un campo vectorial que indica en cada punto del campo escalar la dirección de máximo incremento del mismo.

Los algoritmos de optimización utilizan los gradientes del espacio de búsqueda para encontrar las zonas promisorias donde se prevé encontrar las mejores soluciones. Estos gradientes proporcionan información importante al algoritmo respecto de la dirección de búsqueda, aunque todas las técnicas exitosas utilizan además componentes estocásticos y operaciones de búsqueda que se rigen parcialmente por la información del gradiente, para evitar convergencias prematuras sobre óptimos locales.

En los algoritmos probabilísticos, la información del gradiente se aproxima utilizando muestreos de puntos del espacio de búsqueda. Esta propiedad permite aplicar este tipo de técnicas de búsqueda a funciones discontinuas, no diferenciables.

Óptimo Local

Un óptimo local es un punto en el espacio de búsqueda que se corresponde con una solución óptima para un vecindario del mismo (puntos A y B de la figura 1.2). La noción de óptimo implica que el dominio de la función (espacio de búsqueda) es un espacio métrico o espacio topológico, para que el concepto de vecindario tenga sentido. Estos puntos tienen las características de ser puntos de atracción para los algoritmos de búsqueda, generando una convergencia de soluciones sobre ellos y atascando el descubrimiento de nuevas soluciones. Un buen algoritmo de búsqueda debe ser capaz de no quedarse atascado prematuramente en óptimos locales.

Existen algoritmos determinísticos y estocásticos que tienen la capacidad de, partiendo desde cualquier punto, trazar una trayectoria entre éste y un óptimo local. Estos algoritmos se conocen como de búsqueda local. La búsqueda de un óptimo local es una procedimiento relativamente sencillo [13].

Existen técnicas híbridas que utilizan un procedimiento de búsqueda general, con capacidad de encontrar las zonas promisorias del espacio de búsqueda, y luego aplican en un nivel inferior un procedimiento de búsqueda local, con capacidad de encontrar valores óptimos en una porción acotada del espacio de búsqueda.

Óptimo Global

Se define como óptimo global al conjunto de puntos del espacio de búsqueda que se corresponden con los valores máximos o mínimos (maximización o minimización) de la función objetivo a optimizar (punto C de la figura 1.2). Estas soluciones son las mejores que se pueden encontrar para el problema en cuestión. La definición de óptimo global requiere que la función objetivo esté definida para todos los valores del dominio, y el espacio de soluciones debe formar un conjunto totalmente ordenado (total order set) para permitir la evaluación de diferentes elementos del dominio de la función [14].

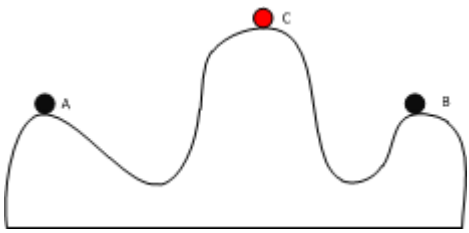


Figura 1.2: Los puntos A y B son óptimos (máximos) locales mientras que el punto C es el óptimo (máximo) global

Conjunto Óptimo

El sub-conjunto de óptimos globales $\mathbf{X}^*$ es el conjunto formado por las soluciones óptimas globales o soluciones con mejores valores para la función objetivo.

$$\mathbf{X}^* \subseteq \mathbf{S} \subseteq \mathbf{X} \qquad (1.12)$$

Función de Probabilidad del Problema

En teoría de la probabilidad, la función de distribución acumulada de una variable aleatoria X es una función que asigna a cada valor real de X la probabilidad de que la misma tenga un valor menor a igual a x. Estas funciones pueden aplicarse también a una variable aleatoria multivariantes (vectores de variables aleatorias) [15].

$$x' \quad F_X(x) = P(X \mathrm{d}x) \qquad (1.13)$$

En este caso, puede definirse como **Función de Probabilidad del Problema**, *Æ*, a la distribución de probabilidad que relaciona cada punto *x* del espacio del problema *X* con la probabilidad acumulada de encontrar al mismo *x* hasta la n-enésima evaluación de soluciones candidatas.

$$Æ(x,n)=P(x \text{ ha sido encontrada hasta la n-ésima evaluación individual}) \quad \forall x \in X,\ n \in N \tag{1.14}$$

Dado que la función *Æ* es una función de probabilidad acumulada, se cumple que

$$Æ(x,n_1) \le Æ(x,n_2) \ \forall n_1<n_2 \wedge x \in X; n_1,n_2 \in N \tag{1.15}$$

$$0 \le Æ(x,n) \le 1 \quad \forall x \in X, n \in N \tag{1.16}$$

Esta definición es interesante dado que incluye todos los componentes del problema de optimización anteriormente descritos. Estos son: el espacio del problema, la función objetivo, las operaciones de búsqueda, la representación y los óptimos encontrados. Por lo tanto incluye tanto el problema a resolver como el algoritmo aplicado a la resolución.

1.3 Algoritmo de Optimización

Puede definirse un algoritmo de optimización como una transformación $(X,F,G,Op,gpm)' \ Æ$ de un problema de optimización en una Función del Probabilidad del Problema *Æ* que encontrará al menos un óptimo local, luego de aplicar un conjunto de operaciones de búsqueda débilmente completo en forma iterativa.

Se puede demostrar que, si se cuenta con un tiempo infinito y el óptimo local existe, el mismo se puede alcanzar.

En este contexto, el mejor algoritmo de optimización es aquel que se relaciona con la Función de Probabilidad del Problema que asigna los mayores valores de probabilidad para el conjunto óptimo $\mathbf{x}^* \in X$ y para los valores más chicos de *n*.

Optimización Local versus Optimización Global

Se denomina **optimización local** a aquellos procedimientos de búsqueda que intentan acceder a los extremos de una función dentro de un vecindario de soluciones candidatas de un punto x^*. Si x^* es un mínimo local de la función $f:\mathbb{C}\subseteq R^n \rightarrow R$, se cumple que

$$\exists\ \acute{}>0 \,/\, f(x^*)\mathrm{d}f(x)\ \ \forall x\in\{x\in\mathbb{C}:|x-x^*|\mathrm{d}\acute{}\} \qquad (1.17)$$

En cambio, se reconoce como **optimización global** a aquellas técnicas pensadas para encontrar los puntos extremos de la función considerando todas las soluciones factibles de la misma. Es decir que si x^* es un mínimo global de una función $f:\mathbb{C}\subseteq R^n \rightarrow R$, se cumple que

$$f(x^*)\mathrm{d}f(x)\ \ \forall x\in\mathbb{C} \qquad (1.18)$$

Como se desprende de la definición anterior, pueden existir muchos óptimos locales que no son óptimos globales. Obviamente, la optimización global es mucho más difícil que la optimización local. Existen pocos métodos para optimización global, en comparación con multitud de métodos para optimización local.

Para el caso de una función continua, dos veces diferenciable, encontrar si un punto es un óptimo local es relativamente trivial. Se testea la primera y segunda derivada en ese punto, si no se encuentran resultados positivos dado que la función es continuamente diferenciable se puede asegurar que un punto vecino puede ser encontrado con un valor menor. De esta forma, es posible crear una secuencia de puntos que convergen a un óptimo local [16].

Continuando con el razonamiento, es posible afirmar que no se puede encontrar el óptimo global de una función f si no se evalúa al menos un punto en cada uno de los vecindarios del punto de referencia. Por lo tanto, cualquier método para resolver un problema de optimización global requiere de una cantidad de pasos no limitados.

Los algoritmos de optimización global son algoritmos que emplean medidas para evitar la convergencia a óptimos locales, e incrementando la probabilidad de encontrar óptimos globales.

Los métodos de optimización global se pueden clasificar en dos clases:

1. Deterministas: no poseen componentes estocásticos, garantizan la convergencia al óptimo global requiriendo necesariamente que la función *f* posea determinadas propiedades [17] [18] [19].
2. Estocásticos: evalúan la función *f* en una muestra de puntos seleccionados al azar y el proceso continúa en forma iterativa evaluando la función en otros puntos. En ningún caso puede asegurarse la convergencia al óptimo global, excepto para la situación cuando la cantidad de puntos evaluados tiende a infinito. Este tipo de procedimiento de búsqueda global no requiere que la función *f* a optimizar cumpla con ninguna propiedad específica, lo que los hace muy versátiles y aplicables a muchos tipos de problemas [20] [13].

1.4 Conclusiones

En este capítulo se definen los conceptos generales asociados a la optimización de funciones matemáticas. A partir de la definición de los diferentes espacios de un problema de optimización, de las restricciones, de la representación de un problema, y de las operaciones de búsqueda se llega a una definición precisa. Ésta definición es de ámbito general, y describe tanto el problema a optimizar como el algoritmo utilizado para encontrar la solución óptima.

Metaheurísticas

Este capítulo se centra en el estudio de las metaheurísticas. Se comienza brindando una definición de esta familia de algoritmos aplicadas al ámbito de la optimización, así como un detalle de los componentes comunes entre todas ellas, incluyendo el manejo de restricciones en problemas de optimización matemáticos. A continuación, se repasan los orígenes de la Computación Evolutiva, profundizando en la técnica más expandida de esta familia, que son los Algoritmos Genéticos. Luego se hace mención de las metaheuristicas más relevantes a lo largo de desarrollo de este campo de estudio. Se incluye una sección dentro del capítulo para nombrar las dificultades que caracterizan a los problemas de optimización, dado que es de mucho interés entender el origen de la complejidad en la búsqueda de soluciones óptimas. Para finalizar, se incluye una sección dedicada a los métodos de paralelización de metaheurísicas, describiendo los avances destacados en el área.

2.1 Introducción

La clasificación primaria de los algoritmos en general, y de las técnicas de optimización en particular, es separar aquellas técnicas exactas (determinísticas) de aquellas técnicas estocásticas (probabilísticas o aproximadas). La categorización respecto del grupo al que pertenece cada algoritmo de optimización no deja lugar a dudas de acuerdo a este criterio, dada la precisa definición de ambos grupos.

Los primeros son aquellos que no incorporan ningún elemento aleatorio, por lo que dada la misma entrada, siempre se obtiene la misma salida.

El segundo grupo incluye aquellos algoritmos que poseen algún componente estocástico. Este componente está relacionado con aumentar la capacidad del algoritmo de explorar ciertas regiones del espacio de búsqueda. En este caso, aún con la misma entrada, el algoritmo devuelve salidas diferentes.

En las siguientes secciones se mencionan las técnicas de optimización más relevantes, profundizando en el grupo de las técnicas estocásticas, mas específicamente las conocidas con el nombre de Metaheurísticas.

2.2 Técnicas de Optimización

La figura 2.1 muestra un cuadro con una posible taxonomía de las técnicas de optimización existentes. En él se refleja el origen de cada una de las técnicas incluidas, así como el orden cronológico asociado con cada una.

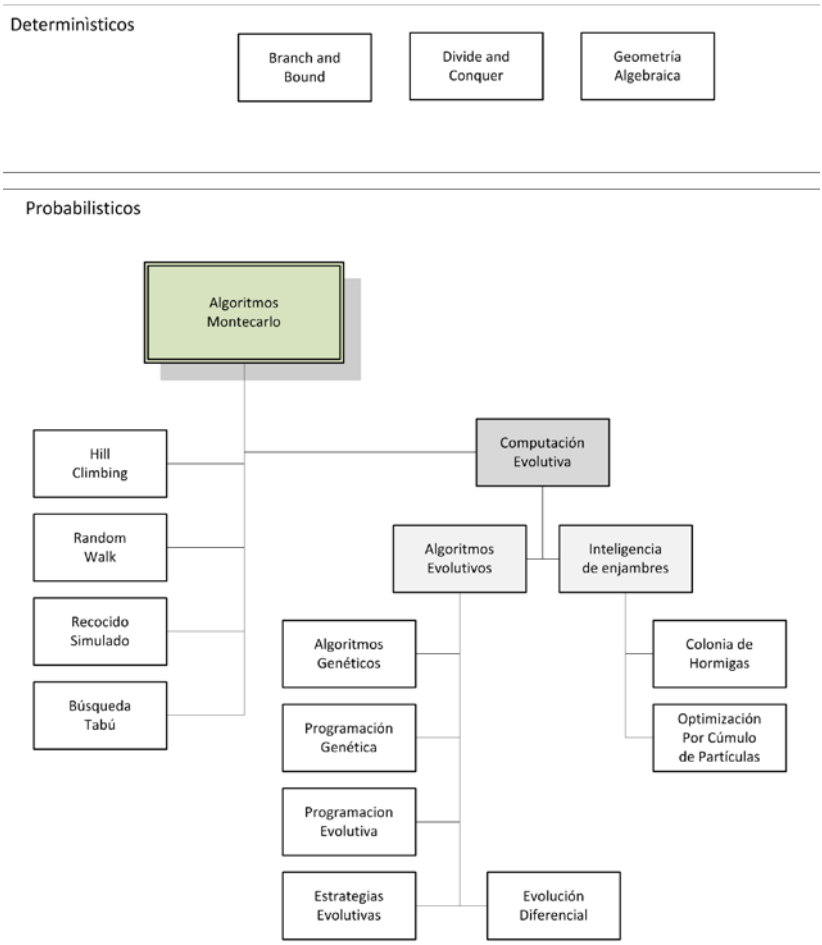


Figura 2.1: Posible taxonomía de las principales técnicas de optimización existentes [8]

En [8] se pueden encontrar detalladas descripciones de los algoritmos y metaheurísticas incluidas en el cuadro de referencia.

Dentro de las técnicas probabilísticas, que son las de interés en la presente tesis, se puede apreciar en la figura 2.1 que existe un punto en común que son los algoritmos o métodos de Montecarlo. Este es un método no determínistico utilizado para aproximar expresiones matemáticas complejas. Su nombre se debe al casino de Montecarlo en una clara referencia a su naturaleza aleatoria. Su origen se remonta a mediados de los 40 cuando Stanislaw Ulam y John von Neumann lo utilizaron en el laboratorio de Los Álamos durante el desarrollo de la bomba atómica.
Desde esta raíz común de las técnicas aproximadas, se desprenden dos ramas claramente diferenciadas. La rama de la Computación Evolutiva, que incluye aquellos algoritmos inspirados en mecanismos propios de la naturaleza, y una segunda rama que agrupa algoritmos estocásticos que utilizan componentes aleatorios para guiar el proceso de búsqueda pero que no se basan en principios de la evolución biológica.

2.3 Definición de Metaheurística

Para entender el término de metaheurísticas, se comenzará dando una definición del término heurística.
Puede definirse como heurística a aquellos aspectos vinculados con la utilización de conocimiento para realizar una tarea. Es decir, a una técnica, método o procedimiento inteligente que no es producto de un análisis formal, si no más bien de conocimiento experto sobre el tema. Las heurísticas son procedimientos específicos, que se aplican en la resolución de problemas determinados, caracterizándose por tener buen rendimiento, aportando soluciones de calidad y poca utilización de recursos.

En Optimización Matemática se usa el término heurístico en contraposición a exacto, que se aplica los procedimientos a los que se les exige que la solución aportada sea óptima y factible.

Una solución heurística de un problema es la proporcionada por un método heurístico, es decir, aquella solución sobre la que se tiene cierta confianza de que es factible y óptima, o de que alcanza un alto grado de optimalidad y/o factibilidad, aunque no se pueda precisar con exactitud cuál es ese grado.

Puede definirse como metaheurísticas a un familia de algoritmos, generalmente estocásticos, de propósito general, que guiados por un heurística recorren en forma iterativa el espacio de búsqueda del problema.

La metaheurísticas son heurísticas de nivel más alto, que define la estrategia principal de guía de búsqueda.[20]

Otra definición dada por Glover es *"métodos que integran de diversas maneras, procedimientos de mejora local y estrategias de alto nivel para crear un proceso capaz de escapar de óptimos locales y realizar una búsqueda robusta en el espacio de búsqueda. En su evolución, estos métodos han incorporado diferentes estrategias para evitar la convergencia a óptimos locales, especialmente en espacios de búsqueda complejos"* [20].

A continuación se describen las principales características de una metaheurística:

- Propósito general: no se relacionan con problemas específicos, si no más bien que son de uso general y aplicable a un amplio conjunto de problemas.

- Fácil implementación: son en general de fácil implementación, dado que normalmente no implican muchas líneas de código, y su amplio espectro de uso las hace naturalmente re-utilizables.

- Fácil paralelización: por su estructura común, es relativamente sencillo definir implementaciones paralelas de las diferentes metaheurísticas existentes.

- Aproximados: se caracterizan por encontrar soluciones cercanas al óptimo, en tiempos de procesamiento relativamente cortos. Muchas veces, ni siquiera puede saberse cuán cerca del óptimo se encuentra la solución hallada. Pese a esta fuerte limitación teórica, se ha demostrado en forma empírica que para aquellos problemas que no pueden resolverse en forma exacta, o que no se cuenta con el tiempo para evaluar todas las soluciones posibles, los resultados ofrecidos por este tipo de algoritmos son de buena calidad y representan la mejor forma conocida de recorrer espacios de búsqueda complejos e inmensos.

- Poca base teórica: generalmente se cuenta con muy poca base teórica, y los estudios en este campo son limitados. Por su naturaleza mayoritariamente estocástica, lo único que se puede afirmar es que la probabilidad de encontrar el óptimo global es igual a 1 cuando la cantidad de iteraciones (evaluación de funciones) es infinito.

2.4 Computación Evolutiva

En el campo de ciencias de la computación, la Computación Evolutiva es una rama de la inteligencia artificial que intenta resolver problemas de optimización combinatoria. El desarrollo original de estas idea se remonta a mediados de los 50, pero es en los 60's donde comienzan tres líneas de investigación en paralelo, John Henry Holland y el desarrollo de los Algoritmos Genéticos, Lawrence J. Fogel con el desarrollo de Programación Evolutiva y en Alemania, Ingo Rechenberg and Hans-Paul Schwefel introdujeron Estrategias Evolutivas.

A principios de los 90's estas corrientes de investigación convergieron a un único campo de investigación agrupándose todas bajo el título de Computación Evolutiva [21] [22] [23].

Unos años después, fines de los 80's, se agregó dentro de este tópico de investigación un conjunto de técnicas bioinspiradas basados en el comportamiento colectivo de agentes autoorganizados. Esta subrama de la Computación Evolutiva adoptó como nombre Inteligencia de Enjambres (*Swarm Intelligence*), nombre tomado del trabajo de Beni [24].

Introducción

La computación evolutiva utiliza normalmente una población que evoluciona a lo largo del tiempo, y donde los individuos más aptos relacionados con el problema actúan como guías del proceso de búsqueda. Estos conceptos están inspirados en principios de la Teoría de Evolución de las Especies de Charles Darwin [24].

En la naturaleza, los individuos compiten por recursos escasos como ser comida, refugio, agua o pareja. Estos individuos se van adaptando al medio que los rodea. Aquellos individuos que presentan una mayor grado de adaptación aumentan la probabilidad de llegar a una edad adulta y, en consonancia, es mayor la probabilidad de tener descendientes.

Este proceso no es de naturaleza determinística, si no mas bien de naturaleza estocástica. Mayor adaptación implica mayor probabilidad de obtener recursos y de descendencia, aunque en ningún caso la asegura. Asimismo, cuando surgen las nuevas generaciones basados en el código genético de sus predecesores, existen mutaciones aleatorias cuya utilidad de acuerdo a proceso de adaptación no se conoce de antemano.

Basados en estos principios de la Teoría de la Evolución surgieron las primeras técnicas de esta rama de la investigación.

Componentes Generales

En este apartado se describen los componentes comunes, siempre presentes, en una metaheuristica aplicada a optimización. Se debe notar que esta sección hace referencia a los componentes conceptuales de una técnica aplicada a maximización o minimización de funciones, sin entrar en los detalles de diseño y/o implementación de alguna técnica en particular. Estos conceptos se repiten en los siguientes capítulos, pero haciendo referencia a procedimientos específicos de las técnicas bajo análisis. La comprensión de los elementos comunes de los algoritmos de esta clase permiten el entendimiento de la raíz que comparten, y a la vez nos facilita el discernimiento de las diferencias existentes entre todos ellos.

Individuo

El concepto de individuo es central cuando hablamos de metaheurísticas. Se denomina individuo a una posible solución del problema. Esta denominación se debe a que una de las primeras técnicas desarrolladas fueron los algoritmos genéticos, inspiradas en la evolución de las especies.

Normalmente, se parte de un conjunto de individuos iniciales (soluciones generadas al azar o con una heurística *ad-hoc*) y estas soluciones se van mejorando (evolucionan) con el correr de las iteraciones del algoritmo, hasta que se alcanza un criterio de parada definido.

El término individuo se utiliza normalmente en aquellas técnicas que evalúan mas de una solución a la vez.

Población

El concepto de población, en metaheurísticas, es similar a la definición en biología. Una población es un conjunto de individuos. En el ámbito de las metaheurísticas, se denomina población al conjunto de individuos correspondiente a una iteración del algoritmo. En el contexto de algoritmos genéticos, se puede llamar también como generación.

En aquellas técnicas de búsqueda que evalúan una única solución a la vez, podríamos decir que sus sucesivas poblaciones están compuestas por un individuo.

Función de Aptitud (*Fitness*)

La función de aptitud (*fitness*) es una función escalar matemática que devuelve un valor numérico para cada punto del espacio del problema.

Esta función debe estar diseñada para que los puntos del espacio del problema que representen las mejores soluciones, tengan los mejores valores de *fitness* y viceversa. Es una medida directa de la utilidad de los puntos del espacio del problema, o de su orden de prioridad / importancia.

Su nombre fue tomado del concepto biológico de aptitud, utilizado en los primeros algoritmos genéticos. El concepto de función de aptitud escalar también se corresponde con el concepto de potencial o función de energía en física. La única diferencia entre ambos conceptos es que los físicos piensan en términos de minimizar la función potencial, mientras que para los biólogos la función de aptitud debe ser maximizada. Matemáticamente esta diferencia es irrelevante, dado que se puede transformar un problema de maximización en minimización o viceversa tomando la inversa ($f(x)*-1$) de la función a optimizar.

Se puede conceptualizar como la implementación de la función objetivo.

Criterios de Terminación

Las metaheurísticas son procedimientos de búsqueda y optimización no exactos, generalmente probabilísticos. Son capaces de encontrar buenas soluciones sin recorrer todas las opciones posibles.

Dada su naturaleza estocástica, es importante definir un criterio de terminación. Los criterios más utilizados son:

- Esfuerzo pre-determinado: se define un criterio de parada a priori, fijando un tiempo o una cantidad de iteraciones / evaluación de funciones pre-acordadas. En pruebas de laboratorio, este criterio es muy útil dado que permite realizar comparaciones justas, evaluando resultados de distintas técnicas luego de la misma cantidad de evaluación de funciones.

  En problemas del mundo real este criterio también es de suma utilidad, dado que permite acotar el tiempo de ejecución del algoritmo al tiempo disponible para resolver el problema. Se debe notar, y esa es una de las ventajas de este tipo de técnicas, que siempre ofrecen una solución al problema. Obviamente, a medida que avanzan la iteraciones, esa solución se va mejorando y refinando.
- Falta de nuevos resultados: define una condición de parada del algoritmo cuando se suceden una cantidad *n* de iteraciones o

evaluaciones de funciones de prueba sin lograr una mejora en los resultados encontrados. En esta condición se dice que el algoritmo está atascado. Esto sucede normalmente cuando se encuentra detenido en un óptimo local.

- Logro de una solución de calidad pre-determinada: se utiliza tanto en problemas de laboratorio como en problemas del mundo real. Cuando se utiliza, el algoritmo finaliza en el momento que se alcanza una solución con un grado de optimalidad definido a priori. Este criterio permite también realizar comparaciones justas entre técnicas diferentes, ya que se pueden medir las cantidades de evaluaciones de función que utilizan dos metaheurísticas diferentes para alcanzar soluciones de la misma calidad.

Convergencia

El concepto de convergencia es, al menos, ambiguo en este campo de estudio. Se utiliza esta palabra tanto para definir la capacidad del algoritmo para alcanzar un óptimo local (global), así como un estado del algoritmo que se caracteriza por concentrar todos los individuos en una pequeña porción del espacio de búsqueda, no necesariamente donde se encuentra un óptimo.

La capacidad del algoritmo de converger a un óptimo local (global); la primera acepción del término; es una característica deseable de toda técnica de optimización.

La concentración de las soluciones en un pequeño sector del espacio de búsqueda; la segunda acepción el término; no necesariamente es una propiedad deseable, dado que generalmente la falta de diversidad de soluciones conducen a que el algoritmo se atasque en un óptimo local de manera prematura.

Manejo de Restricciones

Como se definió anteriormente, muchos problemas de optimización tienen asociado un conjunto de restricciones de igualdad y desigualdad. Una solución es factible sólo si cumple con todas las restricciones definidas para el problema de optimización.
Existen diferentes técnicas de manejos de restricciones desarrolladas. A continuación se presenta un breve recuento de las más utilizadas en el campo de estudio.

Penalidad de muerte:

aquellos individuos que no cumplen con las restricciones del problema, se descartan directamente. En este caso, no se aprovecha la información proveniente de estas soluciones. Esta técnica de manejo de restricciones es eficaz solamente si el tamaño de la región factible del espacio de búsqueda es grande comparada con la región no factible.

Funciones de penalidad:

agrega un término a la función objetivo. Este término penaliza aquellas soluciones que se encuentran en la región no factible. De esta forma, los individuos que se encuentran en las regiones no factibles tienen siempre un valor de función de aptitud peor que cualquier individuo de la zona factible [26].
Esta penalidad es normalmente dinámica, proporcional (lineal, cuadrática, etc.) a la distancia que existe a la solución factible más próxima, pero su valor puede estar influenciado además por variables adicionales (iteración del algoritmo, estadísticas de la población, etc.). Existe mucha investigación en este campo en particular, y sin duda un buen diseño de la función de penalización es un componente central para la optimización de problemas sujetos a restricciones.

Restricciones definidas como objetivos:

estos métodos definen nuevas funciones objetivos a optimizar por cada restricción definida en el problema, de esta forma transforman problemas mono-objetivo en problemas multi-objetivo o problemas multi-objetivo en problemas multi-objetivo con mayor cantidad de funciones a optimizar.
Es importante destacar que a medida que se incrementan las funciones a optimizar, el problema gana en complejidad y en cantidad de soluciones posibles [26].

Desigualdades:

esta técnica se debe a Zakian y es aplicable a problemas con más de un objetivo (multi-objetivo). En este método, se identifican áreas de interés por cada función objetivo en forma de rangos de metas (*goal range*) [27] [28]. Luego, se dividen las soluciones encontradas en tres categorías, de acuerdo a su relación con las diferentes rangos de metas.

1. Soluciones que cumplen con todas las metas de todas las funciones objetivos
2. Soluciones que cumplen con algunas de las metas
3. Soluciones que no cumplen con ninguna de las metas

De esta forma, se direcciona el proceso de búsqueda para aquellas zonas con soluciones que satisfacen en mayor medida los rangos de metas.

Algoritmos Genéticos

Los algoritmos genéticos son las técnicas de búsquedas y optimización más utilizadas dentro del campo de los Algoritmos Evolutivos. Esta técnica es un algoritmo de búsqueda basada en los mecanismos de selección natural y genética natural [22]. Ellos combinan la supervivencia del más apto en forma de representaciones estructuradas de soluciones al problema, llamadas individuos, con mecanismos de intercambio de información para la generación de nuevas soluciones. De esta manera, se le da forma a un proceso de búsqueda basado en la información que se obtiene del problema, y a medida que se recorre el espacio de soluciones posibles se direcciona la búsqueda hacia los valores mejores. Estos algoritmos fueron desarrollados por John Holland, sus colegas y estudiantes en la universidad de Michigan [29].

La principal característica de este método es su robustez. En este contexto, el concepto de robustez se asocia con la capacidad del algoritmo de aplicarse a multitud de problemas diferentes, aún a aquellos problemas donde se cuenta con poco conocimiento a priori. La técnica se caracteriza por ser capaz de encontrar soluciones buenas en un amplio conjunto de problemas [30] .

La algoritmo 1 muestra el pseudo-código de un algoritmo genético simple.

Algorithm 1: Algoritmo Genético Básico

```
Generación de población Inicial (Aleatoria);
while la condición de terminación no se alcance do
    Evaluar la aptitud de cada individuo;
    Seleccionar individuos para reproducción;
    Crear descendencia aplicando recombinación y/o mutación a los individuos seleccionados;
    Seleccionar individuos de la población para reemplazo;
    Insertar la descendencia en la población;
```

A continuación se detallan los elementos principales de un algoritmo genético, procesos que siempre se encuentran presentes en todos los

algoritmos pertenecientes al campo de la Computación Evolutiva, con diferentes nombres e implementaciones. Estos procesos u operadores, como se los llama dentro del campo de estudio, son los bloques básicos de funcionamiento de las técnicas comprendidas dentro del campo de metaheurísticas evolutivas.

Representación

En los algoritmos genéticos las soluciones se codifican como cromosomas. La manera en que se codifican los cromosomas es específica de cada problema, y suele variar entre problema y problema. Estas estructuras pueden ser vectores binarios, vectores de números enteros, números reales, etc. Un cromosoma está compuesto por un conjunto de genes que ocupan un lugar específico, llamado locus. Cada valor posible que puede tomar un gen se lo conoce como alelo. En el contexto de los algoritmos genéticos se conoce como genotipo a esta representación del individuo. Sobre esta representación se aplican los operadores de Recombinación y Mutación que veremos más adelante, lo que implica que los mismos se tienen que adaptar a la representación elegida. Existe además el concepto de fenotipo, que es la forma en que el genotipo se expresa en interacción con su entorno. La morfogénesis es el resultado de decodificar el genotipo para obtener el fenotipo. Este concepto está estrechamente ligado a la siguiente sección.

Asignación de Aptitud

El concepto de aptitud se tomó prestado de la biología, y se debe al desarrollo original de los algoritmos genéticos como primer representante de metaheurísticas bio-inspiradas de uso importante. La aptitud o adecuación (*fitness* en inglés) define la capacidad de un individuo para sobrevivir y reproducirse, y es un concepto primordial en la teoría de la evolución. Este valor es central cuando se hace referencia a la supervivencia del más apto, dado que los individuos con mayores valores de aptitud son aquellos con mayores probabilidades de sobrevivir y reproducirse.

En el campo de la optimización usando computación evolutiva la función de aptitud es una función matemática que asigna un valor escalar a cada individuo de la población. Este valor escalar se corresponde con el objetivo a optimizar. Los valores que devuelve la función de aptitud deben tener una correspondencia con el objetivo a optimizar, y aquellos puntos del espacio del problema que se corresponden a los menores valores de la función objetivo deberán tener los valores de función de aptitud más bajos, y viceversa [31].

Como se mencionó anteriormente, este concepto originario de los algoritmos genéticos se utiliza en todas las metaheuristicas evolutivas, dado que este concepto permite clasificar a los individuos como mejores o peores soluciones.
De esta forma, la mayoría de los algoritmos de optimización estocásticos privilegian las soluciones "más aptas", aquellas con mejores valores de aptitud en función del objetivo a maximizar/minimizar. Asimismo, generalmente estas soluciones participan en mayor medida que el resto en la generación de nuevos puntos en el espacio de búsqueda [22]. De esta manera, en forma iterativa los algoritmos genéticos van recorriendo el espacio de búsqueda en forma inteligente, estimando los gradientes de la función de aptitud a partir de los muestreos de puntos generados por la estrategia de búsqueda.
La función de aptitud recibe como entrada un individuo (genotipo), decodifica el mismo y devuelve uno o mas valores que caracterizan al individuo y la relación con el entorno (fenotipo).

Reproducción

Este operador tiene como finalidad seleccionar, generalmente, un par de individuos de la población para que actúen como padres de los individuos de la siguiente generación. El concepto de padres en este contexto significa que los nuevos individuos de la población se formarán a partir de la combinación de sus elementos, en una clara analogía con el ciclo de reproducción de las especies. La selección de los individuos está directamente relacionada con el valor de aptitud de cada uno de ellos. Al igual que lo expresado en la principal obra de Charles Darwin [25], la probabilidad de selección de un individuo tiene una correlación positiva con su valor de aptitud o de adaptación al entorno. De esta forma, los individuos con mejores valores de *fitness* frente al problema en cuestión tienen una mayor probabilidad de participar en la formación de las nuevas generaciones. Este proceso, aplicándose en forma iterativa a lo largo de la ejecución del algoritmo, permite que las nuevas soluciones que se encuentran (individuos) tiendan a ser cada vez mejores.
En la versión más simple de un algoritmo genético, la selección es implementada como una búsqueda lineal sobre una rueda de ruleta, donde cada slot de la misma representa cada uno de los individuos de la población. El tamaño de cada slot es directamente proporcional a su valor de aptitud [22]. Existen múltiples mecanismos de selección posibles, entre los que se pueden destacar selección por torneos [32], selección por truncamiento [40] y selección basada en ranking [34] [35]. En todas ellas el principio de funcionamiento básico se mantiene, las mayores probabilidades de selección se corresponden con los mejores individuos.

Recombinación

El objetivo final del operador de recombinación (*crossover*) es la generación de uno o dos individuos hijos a partir de los individuos padres seleccionados en el proceso anterior. Este operador de cruza está directamente relacionado con la representación que se utiliza para definir los individuos. En algunas representaciones, es posible realizar cruzamiento de individuos identificando partes del mismo con determinado criterio, e intercambiando las mismas entre los dos individuos seleccionados en el operador de reproducción. Dentro de las técnicas que se aplican en forma directa, se destacan la recombinación de un punto, recombinación de dos puntos, corte y empalme y recombinación uniforme. Existen múltiples variaciones de estos operadores básicos [36] [37] [38].

Otras representaciones no son aptas para un intercambio directo entre las partes de los individuos. Por ejemplo, para individuos cuyos cromosomas que representan la solución tienen un orden, se han generado operadores de recombinación específicos. Entre ellos puede destacarse PMX (*partially-matched crossover*), CX (*cycle crossover*), OX1 (*order crossover operator*), etc. Todos estos operadores tienen como finalidad que los individuos que surjan de su aplicación representen soluciones válidas para el problema en ciernes [39].

Asimismo, para representaciones de individuos directamente como números reales, el operador de recombinación se implementa utilizando diferentes técnicas de selección en los intervalos definidos por los individuos padres [40] [41]. Esta representación ha ganado cada vez más espacio dentro de la comunidad de investigadores, dados los excelentes resultados reportados [42] [43].

Mutación

El operador de mutación tiene como objetivo introducir un cambio aleatorio en el nuevo individuo. Este operador se aplica con determinada probabilidad de ocurrencia [44], y en general afecta a un subconjunto pequeño de la población, aumentando la diversidad de la misma [45]. Este operador tiene la virtud de aumentar las capacidad de exploración del algoritmo, dado que permite realizar búsquedas en zonas aleatorias del espacio del problema. A diferencia de los operadores anteriores, en este caso se intenta darle al proceso de búsqueda una posibilidad para escapar de óptimos locales, generando individuos en porciones del espacio de búsqueda que no se alcanzarían con la mera aplicación de los operadores de reproducción y recombinación [46] [47] [48].

Está demostrada la gran utilidad de este operador en la mayoría de los problemas estudiados, por lo que casi todas las técnicas implementan un operador de esta clase dentro de su rutina de ejecución [49].

2.5 Metaheurísticas Destacadas

Las metaheurísticas de búsqueda y optimización establecen estrategias para recorrer el espacio de soluciones del problema mejorando las soluciones iniciales, normalmente generadas en forma aleatoria, de manera iterativa.

Este tipo de técnicas permiten transformaciones de las soluciones de búsqueda que no sean de mejora (no monótonas), como procedimiento para escapar de óptimos locales.

La principal ventaja de estas técnicas frente a otros métodos es su flexibilidad y robustez, lo que permite aplicarlas a un amplio conjunto de problemas.

A continuación se mencionan las metaheurísticas más relevantes que marcaron el sendero de este prolífico campo de investigación y aplicación.

Programación Genética

Esta técnica metaheurística es muy interesante dado que se plantea el complejo problema de evolucionar programas de computación. El pionero en esta rama fue Lawrence Fogel, quien en 1964 se planteó el desafío de aplicar algoritmos evolutivos al problema de descubrir autómatas de estado finito. La primera aplicación conocida de esta técnica sobre programas de computación representados en formal de árbol data de 1985 [50]. A este autor se le debe la primera representación moderna de un programa de computación procedural en forma de árbol. Esta representación permite una fácil manipulación del código del programa, y permite la aplicación de los operadores básicos de un algoritmo genético. En los 90's, John Koza se convirtió en el principal investigador en esta técnica de optimización[51][52].

La mayoría de los trabajos en el área están basados en esta representación, donde cada nodo del árbol representa una función de operador y cada nodo terminal se corresponde con un operando (figura 2.2). De esta forma es fácil evolucionar las expresiones matemáticas resultantes de la representación seleccionada [53] [54] [55].

Existen asimismo otras representaciones utilizadas con buenos resultados reportados [56].

Con la impresionante mejora de las capacidades de procesamiento de las computadoras, en los últimos tiempos se han logrado mejoras sustanciales en la aplicación de esta técnica a problemas de laboratorio y problemas del mundo real [57] [58] [59].

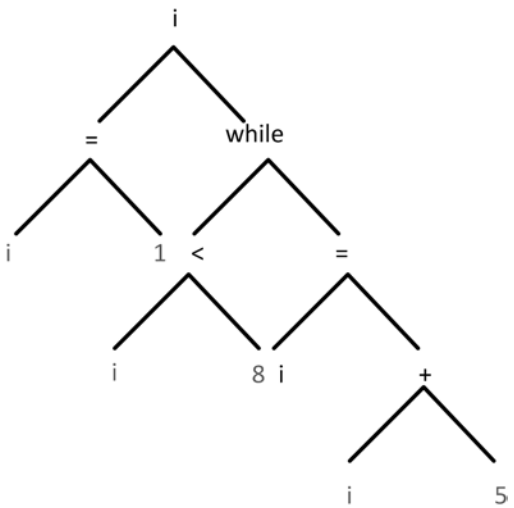


Figura 2.2: Programación Genética. Representación de un programa en forma de árbol.

Estrategias Evolutivas

Esta técnica se desarrolló en los años 60 y 70. La misma utiliza una representación real para los individuos de la población. Es similar a la estructura de un algoritmo genético básico pero utiliza solo los operadores de selección y mutación. Este último operador se implementa seleccionando un valor al azar de una distribución normal generada con cada componente del vector individuo. Existen muchos estudios que indican que uno de los factores centrales que ayudan a la convergencia de los algoritmos a valores óptimos es el valor de la desviación estándar que se utiliza, parámetro conocido como ventana de evolución. Existen diferentes métodos para estimar la misma, aunque en general se trata de rangos de variación pequeños [60].

En su versión canónica la población tiene un sólo individuo, y a partir del mismo se genera un descendiente. Luego, si el descendiente tiene un mejor valor de fitness que el padre, será el elemento seleccionado, en caso contrario, el mismo se descarta [60] [61].

Existen variantes sobre esta propuesta original que agregan más miembros a la población, y algunas con agregados del operador de recombinación. Normalmente, se mejora la capacidad de optimización del algoritmo con estos agregados [62].

Las principales diferencias entre un algoritmo genético (AG) con representación real y una estrategia evolutiva (EE) son:

1. El AG tiende a tener poblaciones más grandes formadas por decenas de individuos.

2. Un AG tienen como principal operador la recombinación, y la mutación es un operador complementario. Una EE tiene como principal operador la mutación, y la recombinación puede llegar a utilizarse como complemento.

3. El proceso de selección de individuos en una AG es estocástico, en cambio en una EE suele ser determinístico.

Evolución Diferencial

Esta técnica fue propuesta por Storn y Price en fines de los 90. Es una técnica pensada para optimización matemática en funciones con dominio real. La misma es extremadamente simple y basa la generación de nuevos individuos en un operador ternario, donde se seleccionan tres individuos al azar, y a la diferencia ponderada de los dos primeros vectores se le suma un tercer vector [63] [64]. Esta simple fórmula de generación de nuevos individuos se ha mostrado como muy eficiente para atacar problemas con representación real [65] [66].

Existen propuestas basadas en el algoritmo original que han mostrado resultados mejores en un vasto conjunto de funciones de prueba [67] [68] [69], así como variantes del algoritmo inicial con procedimientos de auto-adaptación de los parámetros de generación de nuevos individuos [70] [71].

La complejidad computacional intrínseca del algoritmo, sin considerar el costo de la evaluación de la función de fitness por ser propia del problema a resolver, se encuentra entre las más bajas de todas las metaheurísticas.

Swarm Intelligence

La inteligencia de enjambre (SI) es el comportamiento colectivo de agentes descentralizados, que generan sistemas auto-organizados, naturales o artificiales. El concepto se emplea en los trabajos sobre inteligencia artificial. La expresión fue introducida por Gerardo Beni y Wang Jing, en 1989, en el contexto de los sistemas robóticos móviles [72].

Los algoritmos basados en este principio en general están compuestos por agentes simples, que interactúan en forma directa, local, con reglas simples, sin un control centralizado, con interacciones con componentes estocásticos. Esta interacción entre diferentes agentes autónomos termina generando un comportamiento "inteligente", de donde emana un patrón de funcionamiento global que se utiliza para optimización de funciones matemáticas complejas, entre otras aplicaciones. Estas técnicas están inspiradas en la naturaleza, en procesos tales como colonias de hormigas, cardúmenes, bandadas o manadas.

Existen más de una decena de algoritmos de este tipo [73] [74] [75] [76] [77] [78]. Entre las propuestas más destacadas de esta categoría de metaheurísticas se encuentran Optimización por Colonia de Hormigas (ACO) y Optimización por Cúmulo de Partículas (PSO). De la primera, se hace una breve descripción en el siguiente párrafo. La segunda metaheurísticas es el tema principal de esta tesis, y en la segunda parte del presente trabajo se profundiza la investigación sobre la misma.

Optimización por Colonia de Hormigas (ACO)

Este algoritmo fue propuesto por Marco Dorigo en su tesis de Doctorado [79]. El mismo está inspirado en el comportamiento de las colonias de hormigas y su búsqueda de caminos óptimos entre la fuente de comida y su colonia (Figura 2.3). Las primeras versiones fueron aplicadas a encontrar caminos óptimos en un grafo (*connected graph*). Las hormigas artificiales intercambian información en forma indirecta, con la utilización de feromona. Esta última es información estadística asociada a conexiones entre nodos. Los arcos pueden tener un valor heurístico que representa el conocimiento que se tiene del problema [80] [81] [82]. Las hormigas deciden el camino en base a un procedimiento de decisión local estocástico. A su vuelta, dejan rastros de feromonas en las conexiones utilizadas. Existen además procedimientos de evaporación de feromona, cuyo objetivo radica en evitar quedar atrapados en óptimos locales.

Un algoritmo de este tipo presenta las siguientes características:

- Las soluciones de buena calidad emergen como un comportamiento global del conjunto de todas las hormigas.
- Para definir qué camino toma, cada hormiga hace uso de información local de ese nodo y utiliza un componente estocástico.
- Las hormigas se comunican en forma indirecta, a través de los rastros de feromona en el ambiente.
- Las hormigas no son adaptativas, si no que ellas modifican en forma adaptativa la forma que el problema es visto por ellas mismas. [82] [83].

En [10] se propone una extensión de este algoritmo enfocada a realizar búsquedas entre la frontera de las regiones factibles y no factibles del espació de búsqueda, tanto para espacios continuos como discretos. Es muy frecuente que las soluciones óptimas a problemas de optimización se encuentren en estas zonas fronterizas.

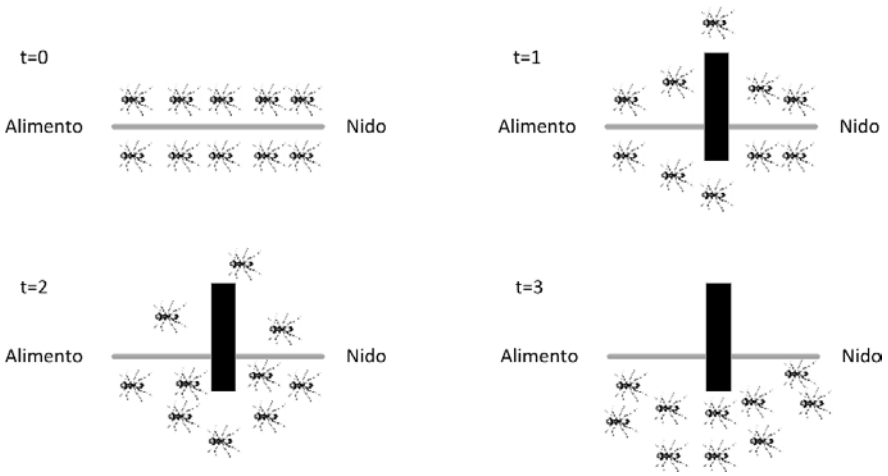


Figura 2.3: Optimización realizada por hormigas en su búsqueda de alimentos

Optimización por Cúmulo de Partículas (PSO)

Este algoritmo fue originalmente diseñado para simular el comportamiento de social de grupos de animales como ser cardúmenes o bandadas (Figura 2.4). La propuesta original se debe a [84], quienes propusieron la primera versión de este algoritmo aplicado a espacios continuos. Poco tiempo después, los mismos autores presentaron versiones aplicadas a problemas binarios [85].

A diferencia de otras metaheurìsticas, esta técnica se caracteriza por almacenar la velocidad de las partículas, ademas de su ubicación. Esto significa que utilizando este algoritmo se cuenta no sólo con las locaciones de las soluciones, sino también con la información de cómo varían estas ubicaciones.

El siguiente capítulo de la presente tesis se dedica a esta metaheurística en exclusiva, por lo que no se ahondará en detalle en esta sección.

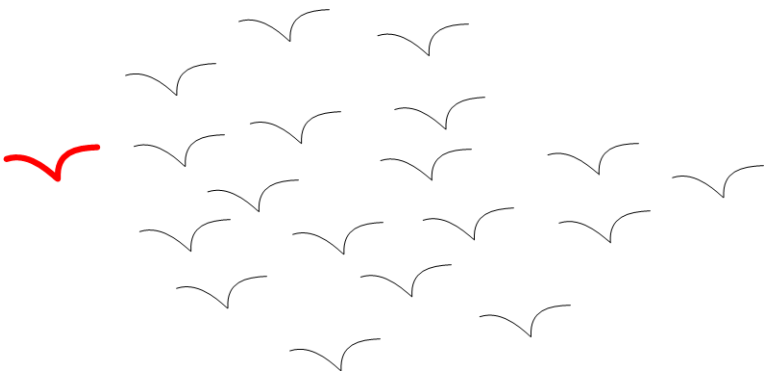


Figura 2.4: Aves en una bandada que responden a un comportamiento basado en seguir a un líder

Hill Climbing

La técnica conocida como Hill Climbing un procedimiento de búsqueda local que intenta encontrar soluciones óptimas cambiando incrementalmente uno de los componentes de la solución. Dada su concepción básica y su simpleza de funcionamiento, esta técnica es muy útil para encontrar óptimos locales. Al igual que las metaheurísticas más avanzadas, este algoritmo puede tener un componente probabilístico. Su uso es bastante popular para la resolución de problemas simples o como algoritmo de búsqueda local, utilizado en conjunto con un algoritmo de búsqueda global [86]. Las búsqueda se direcciona siempre siguiendo el

incremento máximo de la función objetivo. Es un algoritmo del tipo "Greedy", lo que lo hace propenso a quedar atascado en óptimos locales.

Recocido Simulado (Simulated Annealing)

Este algoritmo es una metaheurística de optimización global. Está inspirado en los procesos de cocción de metales y cerámicas. La técnica implica un proceso de calentamiento y luego un proceso de enfriamiento controlado. La aplicación de este procedimiento térmico incrementa los cristales y reduce los defectos del material. Por analogía al proceso físico, a partir de la presente solución se genera una nueva en forma aleatoria. La misma es aceptada con una probabilidad que depende de la diferencia en la función de aptitud de ambas soluciones y un parámetro T (enfriamiento) que decrece con el paso del tiempo (Figura 2.5).

Esta técnica fue desarrollada en forma independiente por dos grupos de investigación a principios de los 80 [87] [88]. El rendimiento del algoritmo es dependiente del proceso de enfriamiento (variación de la variable global T) a lo largo de las iteraciones del algoritmo. A su vez, este proceso es dependiente del problema.

Se ha demostrado en forma teórica la convergencia del algoritmo al óptimo global [89]. Esta demostración tiene valor científico, pero no tiene valor práctico dado que para asegurar la convergencia se requiere recorrer todo el espacio de búsqueda del problema.

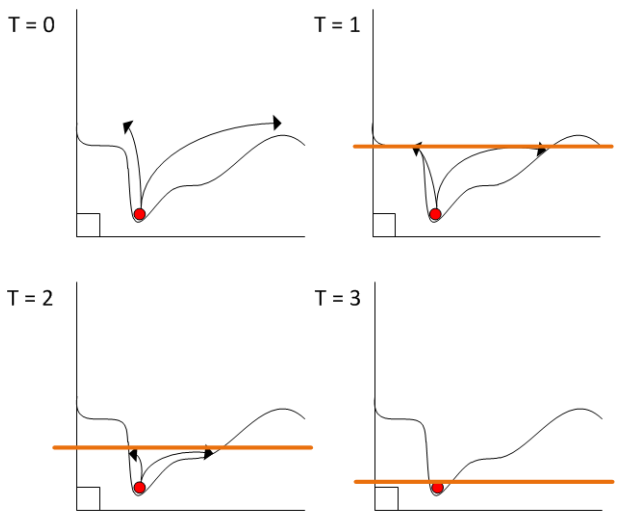


Figura 2.5: Efecto de la variable T en Recocido Simulado

Búsqueda Tabú

Este algoritmo es una propuesta de Fred Glover a fines de los años 80 [90] [91] [92]. Es un procedimiento de búsqueda local que se puede aplicar a problemas de optimización combinatoria. Este procedimiento va recorriendo soluciones vecinas a la actual y utiliza una estructura de memoria con las siguientes características [93] [94] [10]:

- Corto Plazo: almacena las soluciones recientes. Si una solución futura se encuentra en la lista, no puede visitarse nuevamente hasta que alcance el tiempo de expiración.
- Mediano Plazo: lista de reglas que direccionan la búsqueda a las zonas más prometedoras del espacio de soluciones.
- Largo Plazo: reglas que promueven la diversidad, para evitar convergencias prematuras y quedar atrapados en óptimos locales.

Esta técnica se presenta como muy efectiva aplicada problemas discretos.

2.6 Dificultades Encontradas

En esta sección se nombran los principales inconvenientes que se presentan con el uso de metaheurísticas aplicadas a la optimización de problemas.
Estas dificultades de naturaleza general deben ser consideradas al momento de realizar optimización de funciones matemáticas, dado que las mismas explican la mayoría de las situaciones donde se obtienen bajos indicadores de rendimiento.

Convergencia Prematura

La convergencia prematura se refiere a un estado de atascamiento del algoritmo en óptimos locales o en soluciones de mala calidad, en iteraciones tempranas de la ejecución. Cuanto antes se produce, mayor es la pérdida en términos de aprovechamiento de recursos [95].

Como se mencionó anteriormente, encontrar óptimos locales a partir de cualquier punto es una tarea relativamente sencilla para cualquier procedimiento de búsqueda y optimización. Un algoritmo de optimización bien diseñado tiene mecanismos para escapar de máximos o

mínimos locales, y continuar el proceso de búsqueda con el objetivo de encontrar los óptimos globales.

Algunos autores dicen que se puede resumir toda la problemática de la optimización entendiendo el desafío de no quedar atascado en un óptimo local.

La convergencia prematura, normalmente explicada por una excesiva atracción de un óptimo local, es uno de los problemas más frecuentes en el mundo de la optimización.

Existen diferentes componentes para evitar esta condición no deseada. Entre los principales mecanismos utilizados por las diferentes técnicas podemos mencionar los siguientes, aunque todos ellos tienen una raíz común, la incorporación de aleatoriedad en el proceso de búsqueda:

- Variable aleatoria: Todas las técnicas de optimización basadas en metaheuristicas son de naturaleza estocástica. Esto implica la existencia de una o mas variables aleatorias que participan en las fórmulas de los procedimientos de búsqueda. Estas variables aleatorias tienen, en general, una distribución uniforme o una distribución normal. Este componente actúa como reaseguro ante el efecto de atracción que generan los óptimos locales en todas las técnicas de optimización basadas en metaheurísticas. Este componente estocástico se incluye en diferentes tipos de operadores.
- Re-inicio del algoritmo: Existen propuestas donde, al momento de detectarse un atascamiento en el proceso de búsqueda, se reinicia parcialmente o totalmente el proceso de búsqueda. Esta solución, si bien conceptualmente simple, puede dar buenos resultados [96].
- Actualización de ubicación en forma aleatoria: Existen técnicas donde algunos de los individuos de la población toman ubicaciones generadas al azar, ante determinados tipos de eventos o con una frecuencia predefinida. En general, esto se aplica a un subconjunto pequeño de la población. Es conceptualmente similar a la anterior, pero aplicada en forma mas selectiva y esporádica [97].

Deceptividad

Los algoritmos de optimización estocásticos son algoritmos muestrales. Estos procedimientos de búsqueda recorren diferentes puntos del espacio de búsqueda con alguna estrategia determinada, y a partir de las muestras obtenidas se intenta inferir, con diferentes procederes, cual es la “forma”

de la superficie de la función (*fitness landscape*), y así direccionar la búsqueda a estas regiones del espacio de búsqueda [31].
Los problemas deceptivos o engañosos son aquellos donde la forma de la superficie de la función no ayuda a inferir cual es la región donde está el óptimo global, o incluso brinda información que aleja el proceso de búsqueda de esta solución óptima, o lo que es lo mismo, son aquellas funciones donde, para la mayoría de los puntos, el gradiente de la función no apunta a la ubicación de los óptimos globales [98].

Rugosidad (Complejidad del paisaje del espacio de búsqueda)

El concepto de rugosidad está directamente vinculado con la cantidad de óptimos locales. Cuanto mas rugosa un función, mayor es la cantidad de estos máximos o mínimos locales [99]. Cuanto mas máximos y mínimos locales, mayor la dificultad de encontrar un óptimo global. Si a la gran cantidad de óptimos locales le sumamos una superficie de la función con múltiples ascensos y descensos, tenemos una función altamente rugosa [100].

Causalidad Débil

El concepto de causalidad vincula las propiedades de un objeto con su comportamiento. Se denomina causalidad fuerte cuando pequeños cambios en un objeto, producen pequeños cambios en la forma de comportarse. En este ámbito, el concepto de causalidad relaciona las propiedades del individuo con su valor de *fitness*. Se refiere a causalidad débil cuando pequeños cambios en los individuos producen grandes cambios en la función de *fitness*, siendo esta una situación no deseada para un problema de optimización [101].

Los problemas de causalidad débil se pueden deber, en muchos casos, a la selección de una representación incorrecta (mapeo espacio del problema - espacio de búsqueda)

Neutralidad

Se define como neutralidad a grandes regiones del espacio de búsqueda que no presentan ningún gradiente. Estas superficies “planas” no proporcionan ningún tipo de información útil al proceso de búsqueda [102].

Cuando existe muchas neutralidad en un problema de optimización, sucesivas operaciones de búsqueda no modifican los valores de aptitud que se obtienen.

La neutralidad puede ser una característica intrínseca del problema a optimizar o una propiedad presente en el espacio de búsqueda debido a la función de mapeo elegida.

Redundancia

Cuando no existe una relación uno a uno (biyectiva) entre el espacio de búsqueda y el espacio del problema, y un punto del espacio del problema se relaciona con muchos del espacio de búsqueda se dice que hay redundancia. Una solución del problema tiene *n* valores posibles en el espacio de búsqueda con los mismos valores de aptitud (*fitness*). Esta situación provoca la generación de zonas de neutralidad artificiales, producto de las características de la representación del problema [103].

La redundancia es una forma de crear problemas que presentan neutralidad generada en forma artificial, debido a la representación elegida (es decir, mapeo espacio del problema - espacio de búsqueda).
En algunos problemas, tanto la redundancia como la neutralidad pueden ayudar al optimizador en su búsqueda de óptimos globales.

2.7 Paralelización

La paralelización tiene como objetivo reducir el tiempo de ejecución de los algoritmos y/o encontrar nuevas formas de resolución de los problemas. En el ámbito de las Computación Evolutiva, al igual que en las Ciencias de Computación en general se hace referencia a paralelización en los siguientes dos casos:

- cuando se utiliza más de un procesador, o núcleo de procesador, para resolver un problema, o cuando se ejecutan más de una instancia del algoritmo en forma simultánea.

- para hacer referencia a diseños de algoritmos con poblaciones distribuidas espacialmente, independientemente que el algoritmo se implemente utilizando procesamiento en simultaneo.

En [104] existe un inventario detallado de las diferentes versiones paralelas de las metaheurísticas más desarrolladas.
Las técnicas de paralelización pueden distinguirse en las siguientes tres categorías:

1. Paralelización independiente: Esta forma de paralizar implica ejecutar tantas instancias del algoritmo como sean posibles según la capacidad de procesamiento disponible. Al finalizar, sólo se

considera el mejor de todos los resultados obtenidos. Dado la naturaleza estocástica de las metaheurísticas, este técnica tienen sentido para obtener un conjunto de soluciones mas amplio, o para reducir el tiempo de procesamiento de los resultados en investigaciones de laboratorio. Cualitativamente es similar a realizar *n* ejecuciones del algoritmo en forma independiente. Esta técnica no tiene ninguna relevancia desde el punto de vista de la investigación.

2. Algoritmo estándar con distribución del procesamiento: Esta forma de paralelización implica realizar una implementación estándar del algoritmo en cuestión, y aquellas tareas que demandan más procesador realizarlas en forma descentralizada y paralela. Normalmente, la evaluación de las funciones objetivo son las tareas que demandan mayor capacidad de procesamiento. Existen implementaciones de algoritmos que cuando llegan a este paso dentro del hilo de ejecución, distribuyen en *n* procesadores la evaluación de la función de *fitness*, recogen los resultados y continúan con los siguientes pasos del algoritmo. Originalmente, estas implementaciones fueron conocidas con el nombre de Paralelización Global (*global parallelization, farming, master-slave*) donde un procesador central ejecuta las operaciones principales del algoritmo y controla la ejecución general, y delega en *n* procesadores la ejecución de operadores complejos o de la evaluación de la función de aptitud [105]. Estas implementaciones son cualitativamente iguales a la implementación secuencial del algoritmo original, con la única diferencia que se reduce el tiempo de ejecución medido en horas reloj. Los resultados son exactamente los mismos que la versión original de la técnica en cuestión. Es importante distinguir la cuestión del modelo distribuido con la implementación distribuida. En este caso, es un modelo secuencial con una implementación paralela.

3. Algoritmos distribuidos: Esta forma de paralelizar implica la definición de un modelo distribuido, independientemente se efectúe una implementación secuencial del mismo o paralela. Se definen *n* poblaciones distribuidas espacialmente y una población formada por subpoblaciones y cada grupo de individuos intercambia información con el resto de alguna forma predefinida. Pueden destacarse dos metodologías ampliamente utilizadas

    (a) Distribuido (*Coarse Grain*): En esta modalidad existen diferentes poblaciones separadas espacialmente. Comienzan su ciclo de ejecución en forma estándar, y entre

intervalos previamente definidos las diferentes poblaciones intercambian información en forma de individuos aislados u alguna otra forma pre acordada. Este intercambio de individuos permite aumentar la diversidad en cada una de las poblaciones, minimizando la probabilidad de quedar atrapados en óptimos locales debido a convergencia prematura del algoritmo. [105] [106]. Entre los parámetros a definir en este tipo de implementaciones se encuentran:

- La topología de las poblaciones que permite determinar cuales son las poblaciones que intercambiarán individuos.
- La política de migración que permite identificar al individuo que debe migrar.
- El intervalo de migración que indica la frecuencia con la que se intercambian los individuos.
- La técnica de remplazo que especifica a quién remplaza el individuo recibido por cada población.

Es fácil realizar implementaciones paralelas de este modelo distribuido, separando cada población en nodos diferentes de una red, o en procesadores diferentes de una máquina multi-procesador.

(b) Celular (*Fine Grain*) : En esta modalidad existe un nuevo concepto denominado vecindario. Se define de esta forma al conjunto de individuos con los cuales puede interactuar un individuo dado. Los vecindarios tienen cierto nivel de solapamiento. En la figura 2.6 se muestran las topologías de vecindarios más habituales mostrados en una disposición de dos dimensiones. De esta forma, al limitar las posibilidades de selección al alcance definido por el vecindario, se hace más lenta la difusión de los mejores individuos globales, aumentando la capacidad de exploración del algoritmo, y reduciendo la convergencia a óptimos locales. En esta forma de paralelización, la búsqueda de mejores soluciones locales (exploración) ocurre dentro de los vecindarios definidos. Está demostrado que en este tipo de algoritmos los vecindarios tienden a formar nichos que actúan como poblaciones separadas (islas). En esta modalidad se deben definir topologías de

población y vecindarios, cada una de ellas mas adecuada dependiendo del tipo de problema, en función de la relación exploración / explotación que ofrecen [107] [108] [109] [110] [111] [112]. Implementar en forma paralela este modelo es más complejo que el anterior, y normalmente se utilizan threads (hilos) de ejecución diferente por cada individuo [113].

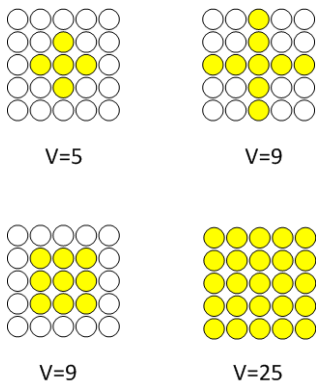


Figura 2.6: Topologías de vecindarios en 2D utilizables en algoritmos distribuidos

Es importante destacar que esa forma de paralelizar genera resultados cualitativamente disímiles de la versión secuencial del algoritmo y por lo tanto de las dos formas precedentes.

Nótese que cualquiera de las dos modalidades de paralelización anteriormente descritas tiene que ver con el diseño intrínseco del algoritmo y no con la cantidad de los procesadores que se utilizan. Es ahí donde reside el mayor valor agregado de estas propuestas. Luego, en ambos casos, dada la naturaleza de los mismos, se define hacer implementaciones paralelas para reducir los tiempos de ejecución [114].

El campo de paralelización es un campo muy fértil en metaheurísticas evolutivas. Para consultar el estado del arte en estos tópicos referir a [104] [105] [111].

Tipos de Optimización

Cuando se enfrenta a un problema de optimización primariamente debe definirse cuáles son los criterios (funciones objetivo) a optimizar. Si se quiere optimizar sólo un criterio, se denomina optimización mono-objetivo. Un ejemplo podría ser el diseño de un nuevo automóvil, y el único criterio a considerar es su eficiencia en el consumo de combustible. En este ejemplo, la solución óptima es el prototipo que presenta el menor consumo de combustible por kilómetro.

Los problemas mono-objetivo están ampliamente estudiados en la literatura, y la gran mayoría de las investigaciones científicas relacionadas con búsqueda y optimización versan sobre problemas con una única función objetivo.

En general, en problemas del mundo real, se optimizan varios criterios en forma simultánea. Siguiendo con el ejemplo anterior, se podría querer minimizar el consumo de combustible así como maximizar el uso de componentes comunes con plataformas desarrolladas en el pasado. Se puede ver que estos dos objetivos son contrarios, para lograr mejores rendimientos en el consumo de combustible normalmente se requiere más innovación, y viceversa. En este caso, el desafío del problema de optimización pasa por encontrar el conjunto de soluciones que optimizan ambos criterios a la vez, aquellos automóviles que presentan la mayor economía de combustible dados los diferentes porcentajes de utilización de componentes comunes, o lo que es lo mismo, aquellos vehículos que para cada nivel de uso de componentes de plataformas existentes sean los más eficientes a nivel consumo de combustible.

Cuando se optimiza más de un objetivo a la vez se denomina optimización multi-objetivo. En este tipo de problemas; a diferencia de la optimización mono-objetivo donde la solución es generalmente única; la solución está compuesta por un conjunto de elementos óptimos, y normalmente se requiere la intervención de un Tomador de decisiones (*Decision Making*) para que seleccione una de ellas de acuerdo a criterios del negocio.

3.1 Optimización Mono-Objetivo

En esta sección se presentan los conceptos básicos de optimización mono-objetivo. Se realiza una definición formal de la misma, y luego se incluye una sección dedicada a la forma de medir el rendimiento de los algoritmos de optimización aplicados a problemas con un único objetivo.

Introducción

Optimizar significa encontrar los valores máximos (maximización) o mínimos (minimización) de una función entre todas las soluciones factibles del problema. En problemas con un solo objetivo, existe una única función a optimizar. Aquel individuo (solución en el espacio del problema) que se corresponde con el valor máximo / mínimo del problema, es la solución al problema de optimización. Formalmente,

$$Minimizar f(x) = y \tag{3.1}$$

sujeto a

$$g_i(x) \geq 0 \quad i = 1,\ldots,q \tag{3.2}$$

$$h_j(x) = 0 \quad j = q+1,\ldots,m \tag{3.3}$$

donde $x=[x_1,x_2,...x_n]$ es el vector de variables de decisión, $f: x \in R^n \rightarrow R$ es la función objetivo a minimizar,

$$h_j: R_n \rightarrow R, i = 1,\ldots,q, j = q+1,\ldots,m$$

, son las restricciones del problema, que definen las zonas factibles y no factibles del espacio de búsqueda.

El algoritmo de búsqueda y optimización que se utilice para abordar el problema deberá tener la capacidad de encontrar estos puntos extremos de la función, o al menos soluciones que se correspondan con valores cercanos al óptimo.

Evaluación de Rendimiento

La evaluación del rendimiento de los algoritmos de optimización posee un rol central en las actividades de investigación en este tópico. Existe una rama en particular dentro del campo de estudio que se dedica a los temas relacionados con evaluación de desempeño. Entre ellos pueden destacarse:

Investigaciones acerca de la evaluación de resultados

Los trabajos en este ámbito se relacionan tanto con las herramientas para medir el desempeño de los algoritmos, así como con la forma de presentar los resultados.

Al medir el desempeño de una técnica metaheurística en forma comparativa con otra puede considerarse el costo de la solución (*solution cost*) o el tiempo de ejecución (*run-time*) como indicadores de performance de las técnicas bajo análisis.

El primero mide la calidad de la solución o soluciones alcanzadas con determinado esfuerzo de procesamiento, definida esta última de manera justa [115]. En este campo, la unidad de medida para el procesamiento es la cantidad de evaluaciones de funciones de aptitud (*fitness*). El segundo mide el tiempo de procesamiento requerido para alcanzar soluciones con determinada calidad definida a priori.

Es importante destacar en este punto la naturaleza probabilística de los algoritmos de este tipo, dado que normalmente se encuentra un componente estocástico en cada uno de ellos. De esa forma, se tiene que considerar a estas variables (costos de la solución - tiempo de ejecución) como variables aleatorias. Por lo tanto, se debe utilizar la estadística descriptiva para visualizar, analizar y validar los resultados. La estadística ofrece las siguientes ventajas:

1. Un marco de trabajo sistemático para la recolección y evaluación e información (diseño de experimentos) que maximiza la objetividad y la replicabilidad de los experimentos

2. Un fuerte base matemática (análisis estadístico) que provee una medición de eventos probabilísticos sobre la base de inferencias sobre la información empírica [116].

No obstante, en el campo de las metaheurísticas se encuentra cierta resistencia a la realización de experimentos bien diseñados [117].

Pueden distinguirse dos modelos de medición de rendimiento:

1. Univariado: se considera sólo una de las variables (costos de la solución - tiempo de ejecución) como indicador de rendimiento

2. Multivariado: en este modelo, ambas variables son de interés para la evaluación de desempeño.

Para el primero de los modelos donde nos interesa sólo una de las variables, existen dos tipos de pruebas estadísticas inferenciales: paramétricas y no paramétricas. La mayoría de las pruebas paramétricas se basan en el supuesto central que los datos están distribuidos normalmente. Muchos estudios existentes sobre la naturaleza de las distribuciones de los valores de las variables, costo de solución y tiempo de ejecución, parecen indicar que esta suposición no se cumple en el análisis de metaheurísticas, en cuyo caso, las distribuciones son desconocidas. Sin embargo, esto no descarta el uso de estadísticas paramétricas. Es bien sabido que algunas pruebas como las basadas en el ANOVA, son muy robustas a desviaciones de la suposición de normalidad, especialmente para grandes conjuntos de datos. Otras técnicas como la transformación de datos puede ayudar a que los mismos puedan cumplir con el supuesto de normalidad. Pruebas no paramétricas, como las basadas en ranking [118] y en permutación [115] pueden prescindir de la hipótesis de normalidad [119] y entregar resultados correctos.

Es una buena práctica en evaluación de performance de algoritmos probabilísticos basados en una única variable, realizar primero un test de normalidad de la variable elegida. Si la misma satisface el test para todos los resultados, se puede utilizar una prueba paramétrica. Si los tests de normalidad dan negativos para al menos alguno de los resultados, se debe utilizar una prueba no paramétrica. Es importante destacar la robustez de las pruebas no paramétricas, dado que si evaluamos los resultados de dos variables aleatorias con distribución normal, las conclusiones alcanzadas con las pruebas no paramétricas siguen siendo válidas, aunque podrían ser menos exactas.

Para el segundo de los modelos, donde existe más de una variable para medir la performance, el escenario de análisis es más complejo [120], dado que en este escenario no existe un conjunto de resultados totalmente ordenado (*total order set*). Como se hace mención en [121] esto configura un problema multi-objetivo, tema del que se hablará en extenso en la siguiente sección.

La gran mayoría de los resultados en este tópico se evalúan con la utilización de modelos univariados. Estas investigaciones científicas tienen que incluir los siguientes elementos:

1. Una cantidad significativa de corridas de cada algoritmo con cada problema, para generar una muestra de resultados a analizar. Cuando mayor es el tamaño de la muestra (cantidad de corridas del algoritmo), más representativa es la muestra de resultados obtenidos.

2. Una medida de tendencia central, generalmente mediana o media aritmética, para resumir en un único número el resultado de las observaciones (diferentes corridas), en el caso de distribuciones normales.

3. Una medida de dispersión (varianza, desvío medio, rango intercuartil) para medir la desviación entre los resultados y la medida de tendencia central, en el caso de distribuciones normales.

4. Una prueba no paramétrica para aquellos casos de distribuciones donde la hipótesis de normalidad no se cumple.

5. Un test de hipótesis para medir la significancia estadística de los resultados, normalmente utilizando un nivel de significatividad del 0,05.

6. Diagramas de caja (o similar) para mostrar gráficamente el desempeño de las muestras generadas, visualizando el rendimiento comparado entre los diferentes algoritmos [122].

Generación de funciones de prueba

Esta rama de la investigación se dedica a la generación de conjuntos de funciones matemáticas de pruebas (*benchmark functions*) con el objetivo de evaluar la performance de los procedimientos de búsqueda en problemas de naturaleza variada.

Un requerimiento esencial de las investigaciones en este campo es realizar evaluaciones justas y sobre un conjunto de funciones de prueba diverso, para medir el desempeño bajo diferentes condiciones.

Es una buena práctica utilizar un conjunto de funciones variadas. De esa forma se evita que un algoritmo que sepa explotar alguna propiedad en particular se destaque por sobre los demás.

A continuación se detallan características deseables para un conjunto de funciones de prueba, sean estas para problemas mono-objetivo o multi-objetivo

1. Funciones unimodales
2. Funciones multimodales
3. Funciones con óptimos globales en la región central
4. Funciones con óptimos locales en la región central
5. Funciones con óptimos globales en los límites del dominio
6. Funciones con óptimos locales en los límites del dominio
7. Funciones rotadas
8. Funciones con el centro desplazada
9. Funciones con diferentes entornos para los puntos óptimos
10. Funciones no continuas
11. Funciones no diferenciables

Muchas de las funciones generalmente utilizadas tienen propiedades que han sido aprovechadas por algoritmos para obtener excelentes resultados [123] [124]. A continuación se enumeran características de las funciones de laboratorio que pueden ser explotadas fácilmente por algoritmos de optimización, y por lo tanto deberían evitarse:

1. Óptimos globales que tienen los mismos valores en las diferentes dimensiones: existe un conjunto amplio de funciones de prueba que son simétricas. Algoritmos que copian valores de una dimensión a otra son capaces de encontrar estos óptimos fácilmente.
2. Óptimo global en el origen: existen operadores especiales que aprovechan esta propiedad común de las funciones de prueba [125].
3. Óptimos globales en el centro de los espacios de búsqueda: existen operadores que promedian las distintas dimensiones de las variables de decisión entre distintas soluciones. Si los algoritmos empiezan con una distribución uniforme y aleatoria de las soluciones, estos operadores tienden a que la población converja al centro del espacio de búsqueda [126].
4. Óptimo global en los límites: es frecuente en muchos algoritmos, sobre todo multi-objetivos, que cuando los valores de algunas de las dimensiones se van de rango, se considera como valor para esa dimensión el valor límite [127]. De esa forma, si el óptimo se

encuentra en estas regiones, es fácil de encontrar. No obstante, si cerca de estas regiones existen óptimos locales, es muy probable que el algoritmo converja a estas soluciones sub-óptimas.

5. Óptimos locales que se extienden a lo largo de los ejes de coordenadas o ninguna vinculación entre las variables o dimensiones: la mayoría de las funciones de referencia, funciones con muchas dimensiones, siempre tienen una estructura de grilla simétrica y los óptimos locales siempre están a lo largo de los ejes de coordenadas. Esta información podría ser utilizada para localizar el óptimo global. Existen algoritmos que utilizan estas propiedades para localizar el óptimo global rápidamente [125] [128] [129].

Se encuentran numerosos trabajos en la literatura con propuestas de funciones de prueba lo suficientemente variadas para permitir realizar evaluaciones de desempeño comparativa entre algoritmos. Dentro del ámbito de la optimización de funciones con dominios reales podemos destacar los trabajos [130][131][132], que se corresponden a problemas de optimización mono-objetivo con y sin restricciones. En el campo de la optimización multi-objetivo, se destacan los trabajos [133] [134] [135].

Existen también funciones de prueba para problemas de un dominio específico como ser el problema del viajante de comercio, el problema de la mochila, coloreo de grafos, SAT, etc. [136].

El objetivo final es siempre el mismo, generar un conjunto de prueba suficientemente amplio que permita discernir a los investigadores cuáles son los algoritmos que mejor se desempeñan.

3.2 Optimización Multi-Objetivo

Introducción

Los problemas de decisión en la vida real involucran, la mayoría de las veces, múltiples criterios de evaluación. Generalmente, estos criterios están en conflicto. Por ejemplo, con el presupuesto acotado de gastos de supermercado, se intenta comprar los productos necesarios de mayor calidad al menor precio posible. Cuando se decide el lugar para vacacionar, seguramente se consideran cuestiones de distancia al lugar, costos de las diferentes alternativas, opciones de entretenimiento para los diferentes integrantes del grupo, extensión del período de vacaciones, etc. Estos dos ejemplos del mundo real permiten entender la naturaleza básica de la optimización multi-objetivo. Se trata de la búsqueda de opciones

considerando la optimización de varios objetivos en simultáneo y normalmente contrapuestos.

Un problema multi-objetivo difiere de un problema mono-objetivo, dado que los primeros requieren la optimización simultánea de más de una función objetivo en paralelo.

La noción de **óptimo** tiene que ser redefinida en este contexto, donde en lugar de buscar una única solución mejor, se intenta producir un conjunto de buenas soluciones de compromiso. El desafío principal de los algoritmos de optimización multi-objetivo es encontrar este conjunto de soluciones para ofrecer al tomador de decisiones (*Decision Maker - DM*) las mejores alternativas entre las disponibles, para que este último seleccione una de ellas [137].

Teoría de la Decisión Multi-criterio

(MCDM - Multi Criteria Decision Making)

Esta disciplina brinda soporte a los tomadores de decisiones cuando se enfrentan a problemas complejos, que requieren de numerosas y generalmente conflictivas evaluaciones. El objetivo perseguido es eliminar la subjetividad en el proceso de decisiones, generando procedimientos de evaluación de alternativas profesionales. La Teoría de la Decisión Multi-criterio es una rama de la Investigación Operativa que en forma explícita considera criterios múltiples en el proceso de toma de decisiones. Es muy habitual en cualquier proceso decisorio que se deban considerar criterios contrapuestos. Analizar problemas complejos desde esta perspectiva colabora a la compresión de la situación y permite entender los criterios de fondo que fundamentan la decisión. Si tenemos mas de un criterio que forma parte del proceso de evaluación de soluciones posibles, necesariamente tiene que haber mas de una solución viable. En este escenario, para obtener la mejor solución al problema implica que debemos obtener las preferencias de tomador de decisiones en función de los múltiples criterios. La forma de articular el rol del Tomador de Decisiones y la consideración de su escala de preferencias es un tópico central en este campo de estudio.

La primera referencia MCDM se asocia a Benjamin Franklin, quien ante un asunto muy importante le escribió una carta a un amigo, Joseph Priestly, donde detalló de una lado los argumentos a favor y del otro los argumentos en contra, y luego vinculó los de mayor importancia de cada lado, en un proceso de ponderación [138].

Es de destacar el trabajo del Marques de Condorcet, quien fue el primero en aplicar procesos matemáticos a las ciencias sociales, en particular a los

procesos eleccionarios [139]. Este trabajo describe la conocida Paradoja de Condorcet, que demuestra que las preferencias colectivas son cíclicas (no transitivas) aunque las preferencias individuales no lo sean.

El desarrollo de la teoría de conjuntos por parte del matemático alemán nacido en Rusia Georg Cantor tuvo un impacto directo en los fundamentos matemáticos que rigen la optimización multi-objetivo.

Por último, son de destacar los trabajos en este campo del matemático Wilfredo Pareto, quien fue el primero que estudió la agregación en un índice compuesto de objetivos en conflicto [140]. Este autor fue el primero en introducir el concepto de eficiencia, conocido como optimalidad de Pareto, definición central en MCDM. Dada una asignación inicial de bienes entre un conjunto de individuos, una asignación de recursos Pareto-óptima implica que no es posible beneficiar más a una persona sin perjudicar a otra.
Dentro de esta rama de investigación, los métodos se dividen en:

Multiple-criteria evaluation problems:

ésta rama agrupa aquellos métodos que apuntan a seleccionar la mejor decisión entre un grupo de soluciones posibles, ya conocidas a priori. En este caso, sólo se evalúan comparativamente las soluciones, pero las mismas son conocidas al principio del proceso.

Multiple-criteria design problems (multiple objective mathematical programming problems):

éste campo de estudio se centra en aquellos métodos que apuntan a encontrar las soluciones posibles al problema, lo que implica una optimización de un modelo matemático. Claramente, las técnicas metaheuristicas se encuentran dentro de este campo de estudio.

Espacio Objetivo

Las dimensiones del espacio objetivo se corresponden con la cantidad de funciones a optimizar. En problemas mono-objetivo este espacio es unidimensional, dado que cada vector de decisión sólo se corresponde con un número escalar. En problemas multi-objetivo, este espacio es multi-dimensional, donde cada dimensión se corresponde con cada función objetivo a optimizar.

Es importante la distinción entre el espacio de decisiones o espacio de búsqueda y el espacio objetivo. El primero corresponde al dominio de la función o funciones a optimizar, y generalmente es multidimensional. El

segundo corresponde al co-dominio de la función o de las funciones. El espacio objetivo tiene tantas dimensiones como funciones se optimizan en forma simultánea (figura 3.1).

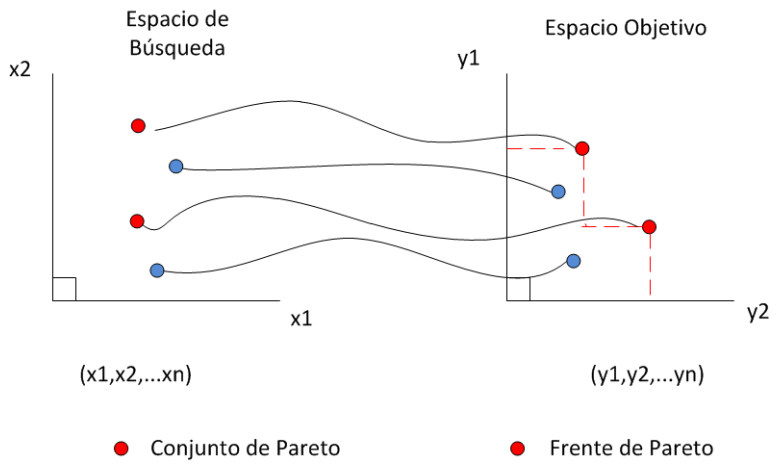


Figura 3.1: Espacio de búsqueda y espacio objetivo. Maximización de funciones

Dominancia de Pareto

En un problema de optimización debe tenerse una medida que permita identificar que tan buenas son las soluciones halladas. En un problema mono-objetivo, la medida en cuestión es el resultado devuelto por la función a optimizar para cada punto solución del espacio de búsqueda (variables de decisión). Dado que esta función devuelve un valor escalar, es posible definir un conjunto totalmente ordenado (*total order set*) de soluciones. Dadas dos soluciones cualesquiera **a** y **b** se puede concluir que:

1. **a** es mejor que **b**,
2. **b** es mejor que **a**,
3. **a** y **b** son igual de buenas.

En un problema multi-objetivo no se tiene una única función a optimizar, sino más bien un conjunto de *n* funciones. De esta forma, para cada solución posible del espacio de búsqueda se obtiene un vector de variables n-dimensional. En estos casos, para evaluar la bondad de las soluciones se utiliza el criterio de dominancia de Pareto.

Este criterio se le debe al matemático italiano Vilfredo Pareto, y permite identificar el conjunto de soluciones pareto-óptimas. Una solución **a** pertenece a este conjunto si no puede encontrarse una solución **b** tal que mejore uno de los objetivos sin empeorar al menos uno de los otros (figura 3.2).

Formalmente,

$$Minimizarf(x):=[f_1(x),f_2(x),....f_n(x)] \qquad (3.4)$$

sujeto a

$$g_i(x)\mathrm{d}0i=1,\ldots,q \qquad (3.5)$$

$$h_j(x)=0j=q+1,\ldots,m \qquad (3.6)$$

donde $x=[x_1,x_2,...x_n]$ es el vector de variables de decisión, $f:[f_1(x),f_2(x),....f_n(x)]:R^n$' R son las funciones objetivo a minimizar, g_i, $h_j:R^n$' $R,i=1,\ldots,q,j=q+1,\ldots,m$, son las funciones que representan las restricciones del problema y que definen las zonas factibles y no factibles del espacio de búsqueda.

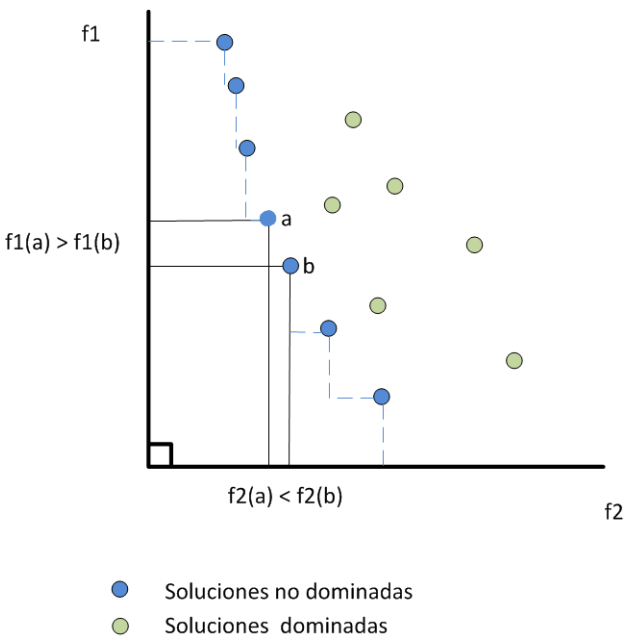


Figura 3.2: Frente de Pareto y Dominancia. Minimización de las funciones f1 y f2

Es importante notar que el concepto de dominancia de Pareto define un conjunto parcialmente ordenado (*parcial order set*) de soluciones. Dadas dos soluciones cualesquiera **a** y **b** se puede concluir que:

1. **a** es mejor que **b**,
2. **b** es mejor que **a**,
3. **a** y **b** son igual de buenas,
4. **a** y **b** son incomparables.

Esta condición del conjunto de soluciones tiene implicancias importantes en los procesos de búsqueda, y obviamente explica porqué el resultado de una optimización multi-criterio está formado por un conjunto de soluciones óptimas.
Está demostrado que el conjunto de soluciones óptimas puede crecer exponencialmente con el crecimiento de las dimensiones del espacio objetivo [141] [142]. Este efecto es de suma importancia y no puede ser ignorado cuando se trabaja con problemas con muchas funciones objetivos (más de 10, por ejemplo). Así, muchos algoritmos de búsqueda que logran excelentes resultados sobre problemas con 2 o 3 funciones objetivo, tiene un desempeño peor que un camino aleatorio (*random walk*) cuando la cantidad de funciones objetivo se incrementa por encima de 10.

Conjunto de Pareto

Se denomina conjunto de Pareto al conjunto de puntos en el espacio de búsqueda que se corresponden con las mejores soluciones al problema de optimización. Este conjunto de puntos puede tener una cantidad de miembros muy alta, creciendo la misma en forma exponencial a medida que aumenta la cantidad de funciones objetivos involucradas. Cada punto del conjunto de Pareto se corresponde con un punto de frente de Pareto en el espacio objetivo (figura 3.1).

Frente de Pareto

Se denomina frente de Pareto al conjunto de soluciones óptimas en el espacio objetivo.
Este conjunto de soluciones se integra con aquellos puntos encontrados en el espacio objetivo que se corresponden con las soluciones no dominadas, de acuerdo al criterio de dominancia de Pareto (figura 3.1).

Este conjunto de soluciones representan las mejores soluciones de compromiso considerando todos los objetivos en juego. Naturalmente,

cuando se la gráfica en el espacio objetivo forma un frente, de ahí su nombre característico.

Este frente puede ser lineal, cóncavo, convexo, continuo, discontinuo, etc., dependiendo de las funciones objetivo integrantes del problema.

Todas las soluciones pertenecientes al frente de Pareto son igualmente buenas, y no se puede especificar si alguna de las soluciones es preferible a las otras, excepto en aquellos casos en que el DM haya definido una preferencia a priori. El desafío de los algoritmos de optimización multi-objetivo se encuentra en encontrar el frente de Pareto real del problema en ciernes, o lograr la mejor aproximación del mismo sujeta a las limitaciones de recursos (tiempo y memoria) existentes.

Tomador de Decisiones (DM - Decision Making)

El rol del tomador de decisiones adquiere relevancia en un problema con más de un objetivo. Los problemas multi-objetivos se caracterizan por tener muchas soluciones posibles, todas ellas igualmente válidas. Cuando mayor es la cantidad de objetivos, mayor es la cantidad de soluciones. Normalmente, el proceso de negocio requiere que se seleccione solo uno de estos posibles cursos de acción.

De esta forma, existen muchos trabajos efectuados para vincular la DM con el proceso de búsqueda de soluciones óptimas, y el momento en el tiempo donde interviene el tomador de decisiones.

Con preferencias definidas a priori en forma de una función de utilidad, es posible convertir el problema multi-objetivo a uno del tipo mono-objetivo, y de esta forma brindar una única solución al DM. Este tipo de intervención del Tomador de Decisiones en el proceso de búsqueda es el menos interesante en las líneas de investigación actuales, dado que la principal ventaja que tiene, que es la simplificación del problema reduciendo los objetivos a optimizar a uno, es a su vez la principal desventaja, dado que trae aparejado una pérdida significativa relacionada con la compresión del problema de optimización. Este tipo de técnicas permite automatizar el proceso de decisión en forma completa, lo que a veces es un requerimiento del problema abordado, vinculado con procesos que requieren encontrar soluciones en cortos periodos de tiempo, y una respuesta en tiempo real (*real-time*). Naturalmente, éste fue el primer abordaje realizado a los problemas multi-objetivo. Como en otras cuestiones de la vida, seleccionar una opción con preferencias definidas a priori, sin profundizar el análisis en otras alternativas disponibles, no asegura en ningún caso una comprensión profunda del problema ni de la solución adoptada [143] [144].

El mecanismo más complejo de interacción entre los procesos de búsqueda y el Tomador de Decisiones pasa por el diseño de un proceso de búsqueda y optimización que considere la participación del DM a medida que el mismo se ejecuta. Este tipo de técnicas definen un proceso de búsqueda muy rico, dado que a la vez que el algoritmo de optimización le brinda información al DM acerca del problema enfrentado, éste último colabora a direccionar el proceso de búsqueda a aquellas regiones del espacio del problema donde considera que se encuentran estas soluciones que minimizan (o maximizan) los diferentes objetivos perseguidos. Como contrapartida, este tipo de procesos requieren más tiempo de ejecución, y el avance sobre el proceso de búsqueda esta determinado además por el grado de participación del DM. Existen diferentes técnicas exploradas en este tópico, con muy buenos resultados reportados. Este escenario, a diferencia del anterior necesita mayor tiempo de procesamiento, más tiempo del DM, y por lo tanto, más tiempo de reloj para encontrar y seleccionar la respuesta [145] [146] [147].

El tercer escenario posible consiste en encontrar todas las soluciones posibles, o un subconjunto de ellas, y ofrecer esta colección de soluciones al DM. Este proceso puede ser tan simple como la visualización simple del Frente de Pareto encontrado, o una visualización asistida con herramientas de apoyo que brinden información de contexto, aunque en ambos casos el concepto que rige la decisión se mantiene. Este tipo de procesos de toma de decisión implican que los algoritmos utilizados para la optimización sean capaces de encontrar frentes de Pareto de alta calidad, y estén preparados para lidiar con n-soluciones en paralelo. Es importante destacar que este último enfoque, que implica la participación del Tomador de Decisiones al final del proceso, aumenta la incidencia del algoritmo utilizado en el proceso de búsqueda, dado que la calidad de la solución final elegida es directamente proporcional a la calidad del frente de Pareto ofrecido por el algoritmo [146] [148] [149].

Cálculo de Aptitud (Fitness)

La calidad de las soluciones, en un problema de optimización, es una característica de relevancia. El concepto de calidad en este campo de investigación es sinónimo al concepto de aptitud, que no es otra cosa que el valor devuelto por la o las funciones objetivo a optimizar para cada punto del espacio de búsqueda. Así como el espacio de búsqueda se corresponde con el dominio de las funciones, el espacio de soluciones se corresponde con el co-dominio.

Este valor de aptitud (*fitness*), término heredado de los algoritmos genéticos, es una medida objetiva que permite identificar entre las

mejores y peores soluciones. En el campo de algoritmos evolutivos, el rol de las mejores soluciones encontradas (líderes) es central, dado que las mismas guían el proceso de búsqueda a aquellas regiones del espacio que lucen más prometedoras. Estas soluciones de alta calidad normalmente tienen asociada una mayor probabilidad de existencia en generaciones posteriores, y actúan como puntos de atracción (atractores) donde terminan convergiendo los algoritmos estocásticos de búsqueda.

En optimización mono-objetivo, la función de aptitud devuelve un valor escalar para cada punto posible del espacio de búsqueda. Esto define un conjunto totalmente ordenado, y permite realizar una selección de líderes libre de ambigüedades, definiendo generalmente el grado de participación de los individuos en la construcción de nuevas soluciones en forma directamente proporcional a su valor de aptitud relativo.

En optimización multi-objetivo ya no se tiene un único valor asociado a cada individuo, si no mas bien un vector de números escalares con una dimensión igual a la cantidad de objetivos a optimizar. Esto define un conjunto parcialmente ordenado, lo que implica que en una comparación entre dos soluciones se de el caso que no se pueda identificar cuál solución es mejor que cuál. Estas soluciones, que no son otra cosa que el conjunto de soluciones óptimas de acuerdo al criterio de optimalidad de Pareto, actúan como líderes influyendo en el proceso de exploración del espacio de búsqueda.

Pese a estas limitaciones enunciadas al momento de identificar la calidad relativa de las soluciones en un problema con más de un objetivo, se han propuesto alternativas originales y efectivas para definir una medida de calidad relativa en la comparación de soluciones no dominadas (incomparables de acuerdo a Pareto). Estas alternativas se basan en una o más de las siguientes propiedades [137] [150]:

1. Cantidad de soluciones dominadas: Este criterio implica que ante dos soluciones cualesquiera (dominadas o no dominadas) se le asigna una calidad superior a aquellas soluciones que dominan una cantidad mayor de individuos en la población. Siempre una solución no dominada es mejor que una dominada.

2. Cantidad de soluciones que la dominan: Este criterio implica que ante dos soluciones cualesquiera (dominadas o no dominadas) se le asigna una calidad superior a aquellas soluciones que están dominadas por una cantidad menor de individuos en la población. Siempre una solución no dominada es mejor que una dominada.

3. Número de frente de Pareto: Este criterio asigna un valor relativo de aptitud de acuerdo al nivel del frente de Pareto donde se

encuentra la solución. De esta forma, las soluciones no dominadas están en el frente 0, y por lo tanto de mayor valor relativo. Luego, se retiran las soluciones del frente 0, y se calcula el nuevo conjunto de soluciones no dominadas. Estas soluciones se encuentran en el frente 1. Luego, se retiran éstas del conjunto y se calculan las soluciones que pertenecen al frente 2. Se continúa con este proceso hasta clasificar todo el conjunto de soluciones.

4. Indicador escalar: Entre dos soluciones no dominadas, se prefiere aquella que presenta el valor más alto (bajo) para un indicador dado. Este indicador se puede calcular utilizando diferentes criterios, pero en todos los casos devuelve un valor escalar asociado a cada solución. El mismo se puede basar, por ejemplo, en el grado de aislamiento de un individuo. En este caso se asigna un mejor valor a aquellos puntos donde las soluciones próximas (vecinos) se encuentran a una distancia euclídea mayor, calculada con algún procedimiento dado. Este criterio de evaluación relativa intenta otorgar mayor valor de aptitud a soluciones en regiones menos exploradas del espacio de búsqueda, y de esa forma incrementar la diversidad de las soluciones encontradas. Otro ejemplo es el cálculo de un indicador que mide el aporte de cada solución a la superficie o volumen final del Frente de Pareto alcanzado. En este tipo de indicadores, siempre se considera en forma directa o indirecta la optimalidad de Pareto asociada con cada solución.

Medidas de Desempeño

Una rama central de este campo de estudio se centra en la propuesta y evaluación de indicadores que permitan medir el rendimiento de nuevos algoritmos de búsqueda, o de nuevas variantes de algoritmos existentes. En la investigación científica se tiene que medir de una forma objetiva, justa y pareja el resultado devuelto por los algoritmos que se proponen.
Un algoritmo multi-objetivo tiene como propósito devolver un frente de Pareto de calidad. Hay tres características centrales que definen la calidad de un frente de Pareto [151] [152]

1. Cercanía: Un frente de Pareto es de mayor calidad cuanto más cerca se encuentre del frente de Pareto real del problema. Se definieron indicadores que calculan esta cercanía de una forma generalmente aceptada. Obviamente, para computar estos indicadores es necesario conocer el frente de Pareto real del problema

2. Distribución: Un frente de Pareto cuyos miembros estén uniformemente distribuidos a lo largo del mismo es de mejor calidad que otro cuya distribución de soluciones es irregular, presentando regiones del frente superpobladas de soluciones, y otras regiones con pocas soluciones o ninguna. Existen indicadores que calculan un valor escalar que representa la uniformidad en la distribución de soluciones a lo largo del frente de Pareto.

3. Cobertura: Este criterio indica que un frente de Pareto es mejor que otro si el grado de cobertura que presenta respecto del frente de Pareto real, es mayor. Utilizando indicadores que miden el nivel de cobertura se intenta que el rango de soluciones encontradas sea completo, y no queden regiones del espacio de búsqueda sin evaluar.

Todos los indicadores de medidas de performance miden, en forma directa o indirecta, los criterios enunciados.

Si bien, desde el punto de vista conceptual, las nociones anteriormente detalladas son claras, la construcción de indicadores que reflejen que un conjunto de soluciones es mejor que otro conjunto de soluciones es una materia compleja.

La necesidad de comparar diferentes salidas obtenidas por dos algoritmos a llevado a la construcción de indicadores unarios que reflejen una medida de calidad de los frentes de Pareto obtenidos por cada técnica. A menudo, se calculan varios indicadores unarios al momento de comparar, para llegar a conclusiones del tipo:

- La aproximación *S* al frente de Pareto es mejor que la aproximación *T*.

Si bien estos indicadores intentan reflejar las características deseables de los frentes de Pareto, en ningún caso podemos afirmar que mejores valores de indicadores de rendimiento para la aproximación *S* respecto de *T* significa que el conjunto de soluciones *S* domina al conjunto de soluciones *T*.

Muchas investigaciones en el área asumen en forma incorrecta que esta propiedad es satisfecha por parte de los indicadores de calidad utilizados.

En [152], Zitzler y otros investigaron los indicadores de calidad y probaron las limitaciones teóricas de los mismos. Entre las observaciones más interesantes podemos destacar:

1. No existe un indicador unario de calidad que sea capaz de afirmar cuando una aproximación *S* al frente de Pareto es mejor que una aproximación *T* al frente de Pareto

2. La definición anterior se mantiene aun considerando una cantidad finita de indicadores

3. Para la mayoría de los indicadores de calidad que indican que *S* es mejor que *T*, en el mejor de los casos sólo pueden indicar que *S* no es peor que *T*, o que ambas son incomparables.

4. Existen indicadores de calidad que podrían se capaces de distinguir que *S* es mejor que *T*, pero su uso está restringido a casos muy particulares.

5. Existen medidas de calidad binarias que resuelven los problemas mencionados, si están diseñadas de forma correcta.

No obstante lo anteriormente expuesto, las medidas de calidad unarias tienen una alta utilidad como indicadores relativos para evaluación de perfomance, aunque se debe tener en consideración lo mencionado.

Como una paradoja curiosa se destaca que evaluar la performance de un algoritmo de optimización multi-objetivo es, a su vez, un problema multi-objetivo, dado que no existe un único indicador que pueda agrupar todas las características deseables de un frente de Pareto.

Archivo de Soluciones Óptimas

Existe un denominador común en la mayoría de las técnicas de optimización multi-objetivo referido al modo de almacenar las soluciones no dominadas que van siendo encontradas. Estas soluciones no dominadas son aquellas que guían el proceso de búsqueda, y que formarán parte del frente de Pareto ofrecido como solución. Estas soluciones se van detectando gradualmente a medida que avanzan las iteraciones del algoritmo, y se almacenan en un archivo externo de soluciones no dominadas. Normalmente, las diferentes técnicas de optimización buscan una solución o más de una de este conjunto para generar los nuevos individuos de las generaciones futuras [153] [154].

Por limitaciones vinculadas con la memoria y/o capacidad de procesamiento, normalmente se limita este archivo externo a una cantidad de soluciones fija, y se utilizan diferentes procedimientos para seleccionar qué soluciones considerar y cuáles descartar cuando la cantidad de soluciones no dominadas supera el límite pre-establecido [150].

Procedimientos de Selección de No Dominadas

Como se describe en el párrafo anterior, la mecánica general en las metaheurísticas multi-objetivo consiste en ir generando en forma incremental un archivo de soluciones no dominadas. Este archivo tiene dos roles principales:

1. Solución al problema: contiene el conjunto de Pareto propuesto por el algoritmo. En el mismo se mantienen aquellas soluciones que de acuerdo a criterios de dominancia no son superadas por otras soluciones. Normalmente se establece que el mismo está acotado a una cantidad fija de soluciones.

2. Guía de búsqueda: los individuos o soluciones que pertenecen a este conjunto actúan como guía primaria de búsqueda durante la ejecución del algoritmo. En general, se selecciona uno o más individuos de este conjunto, algunos individuos de la población, y con intervención de variables aleatorias se generan nuevas soluciones en base a estos tres componentes principales.

En el contexto de estas dos funciones se hace necesario definir criterios de selección de individuos. En el primer caso, el problema de seleccionar soluciones es necesario al momento de elegir qué soluciones descartar cuando el archivo de soluciones no dominadas llega a contener la máxima cantidad permitida. En el segundo caso el problema se presenta al momento de elegir que individuos se utilizarán para generar la siguiente generación de puntos en el espacio de búsqueda. En ambos casos, se considera privilegiar aquellos individuos que se encuentren en zonas menos pobladas del espacio. De esa forma, se consigue una cobertura más amplia del frente de Pareto, característica deseable del conjunto de soluciones [150][155][156].

Normalmente se utilizan procedimientos que calculan factores de nicho (*Niching Methods*), aunque los mismos están asociados con una alta complejidad de cálculo [137]. En [157] se utiliza un procedimiento que genera una cuadrícula en forma de grilla, y posiciona a cada solución en una de ellas. Se mantiene un mapa que indica cuántas soluciones hay en cada grilla, priorizando aquellas soluciones que se encuentran más aisladas. De esta manera, la necesidad de procesamiento se disminuye considerablemente.

En cualquier caso, el objetivo perseguido es encontrar soluciones no dominadas en aquellas regiones del frente de Pareto menos pobladas, y de esa forma aumentar la uniformidad en la cobertura del frente de Pareto del problema.

Propuestas Destacadas en Optimización Multi-objetivo

En esta sección se destacan algunos algoritmos históricos referentes en este campo de estudio. Estas propuestas fueron marcando los senderos de investigación hasta llegar a los algoritmos actuales representativos del estado del arte.

Este compendio resumido nos sirve también como guía para entender el abordaje dado desde la perspectiva algorítmica.

Funciones Agregadas

El primer enfoque, y más natural por cierto, es utilizar técnicas mono-objetivo para resolver problemas multi-objetivo. Para poder reutilizar las técnicas existentes, lo que se hace es transformar el conjunto de funciones objetivos en una única función escalar. Para cada función objetivo se define un factor de ponderación, representado el mismo la importancia relativa de ese objetivo respecto del resto. Un ejemplo más simple de esta técnica es la transformación del vector de funciones objetivo a una función única siendo ésta la suma ponderada de las funciones objetivo cada una multiplicada por el factor de ponderación. A los efectos de normalización, se asume que la suma de los factores de ponderación es igual a 1.

Es importante notar que la superficie (*landscape*) del espacio de búsqueda cambia utilizando diferentes conjuntos de valores de ponderación, con lo que usualmente se abordan estos problemas optimizando la función utilizando diferentes conjuntos de ponderadores.

Este enfoque tiene como principal ventaja la reutilización de las técnicas mono-objetivo existentes, y funciona bien con pocas funciones objetivo y espacios de búsqueda convexos. Como principales desventajas se encuentran la dificultad de encontrar un conjunto de valores de los ponderadores que escale bien cuando se conoce poco del problema, y la incapacidad para generar miembros del frente de Pareto real cuando el espacio de búsqueda es cóncavo [158].

Algoritmo VEGA

Schaffer [159] propuso un algoritmo genético cuya estructura básica es la misma que un algoritmo genético tradicional, y sólo modificó la forma en que se realiza la selección. En cada generación, se generan tantas sub-poblaciones como funciones objetivo existen. Luego, se realiza una selección proporcional considerando sólo el valor de la función objetivo que le corresponde a cada sub-población. Se aplican los operadores

tradicionales de mutación y reproducción, y se genera una nueva generación del mismo tamaño que la original.

Las soluciones generadas por este algoritmo son localmente no dominadas, pero no necesariamente globalmente no dominadas.

La principal ventaja es su facilidad de implementación, dado que se trata de una modificación trivial sobre un algoritmo genético canónico. La principal desventaja está relacionada con el efecto de **especiación**, es decir la generación de especies con muy buenos valores para una sola función objetivo. Esto implica que es muy alta la probabilidad de perder individuos con buenas soluciones de compromiso, que presenten buenos valores de aptitud en todas las funciones, pero que no se correspondan a los mejores valores para ninguna. Estos individuos, aun siendo buenas soluciones globales, se pierden por la naturaleza misma del mecanismo de selección [159].

Algoritmo MOGA

Esta propuesta de Fonseca y Fleming es también un algoritmo genético (*Multi Objective Genetic Algorithm*) [160] donde se elabora un ranking de los individuos de acuerdo a su grado de dominancia de acuerdo a la propuesta de Goldberg [31].

Utiliza también mecanismo de *sharing* y un método de formación de nichos para distribuir los individuos a lo largo del frente de Pareto [161].

Es una técnica de rápida implementación, y generalmente logra buenos resultados. La efectividad del método es dependiente del factor de *sharing* que se utiliza.

Algoritmo NSGA

Esta técnica se corresponde con una implementación de un algoritmo genético que clasifica a los individuos en diferentes niveles de acuerdo al ranking de Pareto [162]. Todos los individuos no dominados se agrupan en una categoría y se les asigna un valor de *fitness* alto. Luego, se aplican técnicas de *sharing* para mantener la diversidad de la población. A continuación, se retira este grupo de la población, y se selecciona el nuevo conjunto de individuos no dominados. Estos obtienen un *fitness* menor que los del grupo inicial. Este proceso continua en forma iterativa hasta que todos los individuos reciben un valor de *fitness* calculados con esta técnica.

Dado que los individuos de los primeros frentes reciben un valor mayor de *fitness*, el algoritmo tiende a realizar búsquedas en las zonas no dominadas.

Algoritmo NSGAII

NSGA II es una versión mejorada de NSGA que apunta a mejorar las críticas realizadas a este último. Las críticas más fuertes recibidas por el algoritmo NSGA estaban relacionadas con los siguientes aspectos: [154].

1. La ordenación y clasificación de individuos utilizada presenta una complejidad computacional alta.
2. No utiliza elitismo. En [163] y [164] se ha mostrado que el uso de elitismo mejora la capacidad de convergencia del algoritmo al frente de Pareto del problema.
3. Es necesario especificar el parámetro de *sharing*.

Particularmente, NSGAII se caracteriza por reducir la complejidad del método de clasificación, aumentando la necesidad de memoria. Asimismo, se le agregó un mecanismo de elitismo y se cambió el procedimiento de *sharing* por uno que compara que tan pobladas están las áreas próximas a cada solución (*crowded operator*). En este algoritmo, la población descendiente *Q* (tamaño N) es creada en primera instancia usando la población de padres *P* (tamaño N). Después de esto, las dos poblaciones son combinadas para formar *R* de tamaño 2N. Después de lo anterior, mediante un ordenamiento no dominado se clasifica la población *R* en diferentes frentes de Pareto de acuerdo a su nivel de dominancia. Una vez el proceso de ordenamiento no dominado ha finalizado, la nueva población es generada. Esta nueva población empieza a ser construida con el mejor frente no dominado (*F*1), continúa con las soluciones del segundo frente (*F*2), tercero (*F*3) y así sucesivamente. Como la población *R* es de tamaño 2N, y solamente existen N individuos que conforman la población descendiente, no todos la individuos de los frentes pertenecientes a la población *R* podrán ser acomodados en la nueva población. Aquellos individuos que no pueden ser acomodados se rechazan (figura 3.3).

Con estos cambios el algoritmo mejoró el desempeño en todos los problemas de pruebas evaluados.

Es de destacar que esta propuesta se considera, aún al día de hoy, como una de las técnicas metaheurísticas de línea base que se utilizan para medir el rendimiento comparativo de las nuevas propuestas.

Se ha utilizado tanto en su versión original como en variantes de la misma para resolver problemas del mundo real con muy buenos resultados [165].

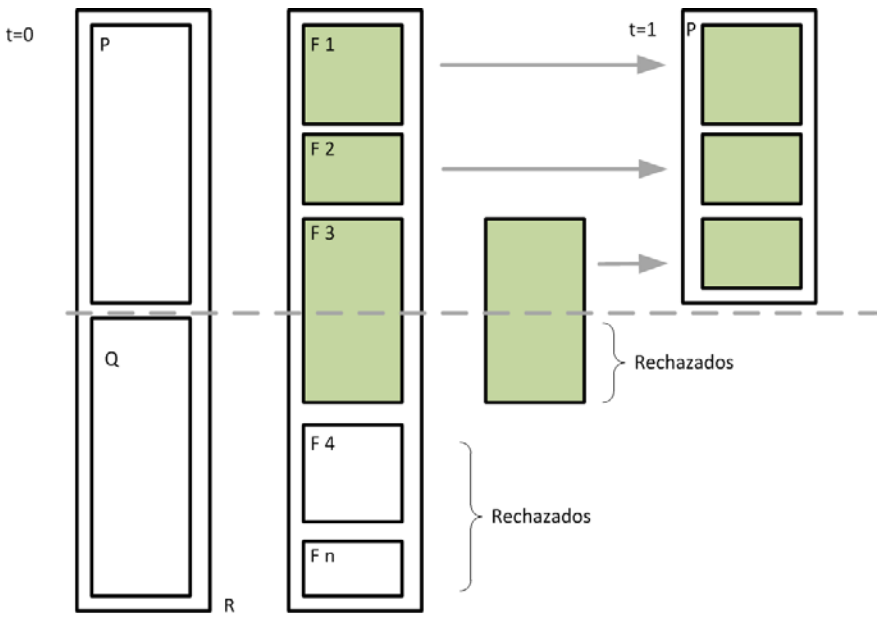


Figura 3.3: Proceso de ordenación de soluciones en NSGA II

Algoritmo PAES

Otra estrategia evolutiva es la técnica conocida como PAES (*Pareto Archived Evolution Strategy*)[157] también conocido como (1 + 1)PAES. Este algoritmo utiliza sólo un individuo como agente de búsqueda, y al igual que la mayoría de las propuestas multi-objetivo, mantiene un conjunto de soluciones externas. Este archivo de soluciones está fundado en una administración basada en grillas. Trabaja dividiendo el espacio objetivo ocupado por los individuos del archivo de soluciones en una cantidad constante de áreas rectangulares, llamadas regiones grillas o cuadrícula de regiones (*grid regions*). El algoritmo intenta encontrar el conjunto de soluciones no dominadas que minimiza el número de

individuos en cada región. De esta forma, se asegura una distribución uniforme a lo largo de toda la extensión del frente de Pareto.

Algoritmo SPEA

Strength Pareto Evolutionary Algorithm es uno de los MOEAs más conocidos. El mismo utiliza dos conjunto de soluciones [166], la población principal donde los operadores evolutivos son aplicados, y un archivo externo con una cantidad variable de soluciones no dominadas. En cada iteración del algoritmo, las mejores soluciones de la población principal se almacenan en el archivo de soluciones externas utilizando una técnica de *clustering*. Luego, se efectúa un torneo binario entre ambos conjuntos como operador de selección. Los individuos que pertenecen a la población externa se les calculan un factor de dominancia similar al algoritmo MOGA. El valor de aptitud de cada individuo en la población principal se calcula considerando el ranking de los individuos de la población externa que este individuo domina [167].

Algoritmo SPEA2

En el año 2001 se propuso una versión revisada del algoritmo SPEA por parte de los autores originales [155]. Esta versión tiene tres diferencias principales con la versión original:

1. Incorpora una asignación de aptitud más elaborada, considerando tanto los individuos dominados por la solución, como los individuos que la dominan.
2. Utiliza una técnica que calcula el individuo vecino más cercano, que mejora el rendimiento del proceso de búsqueda.
3. Se mejoró el proceso de eliminación de soluciones preservando aquellas que están en los límites del frente de Pareto.

Al igual que NSGAII, esta técnica es considerada por la mayoría de los investigadores en este campo como línea base de comparación, por lo que la mayoría de las investigaciones muestran resultados comparativos con la misma.

3.3 NFL Teorema

El teorema *No Free Lunch* (no hay almuerzo gratis) fue presentado por David H. Wolpert y William G. Macready en el año 1995 [168]. En realidad, se trata de un conjunto de teoremas que demuestran que todos los algoritmos de optimización aplicados al conjunto de todos los problemas matemáticamente posibles, se comportan, en promedio, de la misma forma.

Formalmente, un problema es una función objetivo que asigna un valor escalar a cada punto posible del dominio de la función. Un algoritmo de optimización es un procedimiento computacional que genera una secuencia ordenada de puntos visitados con su correspondiente valor asociado.

Por simplicidad, en su análisis Wolpert y Macready estipulan que los algoritmos sólo visitan cada punto del espacio de búsqueda una sola vez, aunque los resultados son igualmente válidos aun sin considerar este supuesto.

Para avanzar con el análisis es conveniente realizar una conversión de los algoritmos estocásticos a los algoritmos determinísticos. Considerando un espacio de búsqueda finito X y asumiendo un conjunto finito Y de todas las posibles funciones objetivo y siendo $|X|$ el tamaño de X e $|Y|$ el tamaño de Y, el conjunto de todas las posibles $F=Y^X$ representa el espacio de todos los posibles problemas, siendo del tamaño de Y^X.

Cada función f se corresponde con cada una de las posibles permutaciones del espacio de búsqueda. Este conjunto es cerrado bajo permutación, y el conjunto de los resultados de búsqueda es precisamente el conjunto de funciones de prueba. De esta forma, todos los algoritmos de búsqueda secuenciales tienen la misma distribución de resultados cuando los aplicamos al conjunto de todas las funciones.

F es el conjunto de todas las funciones objetivos f:X' Y, donde X es un espacio finito de soluciones y Y es un conjunto finito parcialmente ordenado. El conjunto de todas las permutaciones de X es J. Una variable aleatoria R es distribuida sobre Y^X. Para todo j en J, R o j es una variable aleatoria distribuida en Y^X, con $P(R \circ j=f)=P(F=f \circ j^{-1})$ para todo f en Y^X [169].

Digamos que $a(f)$ y $b(f)$ denotan la salida de los algoritmos de búsqueda a y b con la función de entrada f. Si $a(f)$ y $b(f)$ están idénticamente distribuidas sobre todos los algoritmos de búsqueda a y b, entonces F tiene una distribución NFL. Esta condición se mantiene si y solo si F y F o j están idénticamente distribuidas para todo j en J [170].

Es decir, no hay almuerzo gratis (*free lunch*) para algoritmos de búsqueda si y solo si la distribución de funciones objetivos es invariante bajo permutación del espacio de soluciones [171].

Estas conclusiones son similares al teorema del pato feo (*Ugly duckling theorem*) [172], que si bien no es un teorema en su sentido estricto, afirma que una clasificación no es posible sin alguna clase de tendencia.

Las implicancias teóricas del NFL teorema son claras:

1. No existe un algoritmo universal que tenga un desempeño mejor que el resto en el conjunto de todos los problemas posibles.
2. Es necesario incorporar información específica del problema a resolver para lograr un desempeño mejor que el promedio.

No está probado que el teorema NFL se aplique a la clase de problemas *NP*, a menos que se pruebe que *P` NP*. Si *P=NP*, existen algoritmos más eficientes que la búsqueda aleatoria (*random search*)[173].

La utilización de algoritmos estocásticos no hacen nada para mejorar la performance en una corrida única. Agregar aleatoriedad a los procesos de búsqueda sirve para reducir el riesgo de resultados de baja calidad en un conjunto de corridas, sacrificando obviamente la posibilidad de encontrar resultados de calidad superior. Ésta es una estrategia razonable cuando no se conoce la relación entre el procedimiento de búsqueda utilizado y la real distribución de probabilidad del problema [174].

Si bien en la teoría estos resultados son correctos, en la práctica se ha demostrado el buen rendimiento de muchos algoritmos de optimización en conjuntos de problemas grandes. Esta divergencia está explicada porque las funciones objetivo que modelan problemas del mundo real son altamente comprensibles, tienen una estructura y una descripción compacta [175]. Estas funciones comprenden un pequeño subconjunto de todas las funciones posibles. Esta idea sirvió como base para el desarrollo de versiones de NFL más elaboradas, demostrando que en realidad el teorema se aplica a un subconjunto mucho más chico de funciones que lo que se creía en un principio [175].

Las funciones incomprensibles (Aleatoriedad de Kolmogorov) no tienen ninguna regularidad que un algoritmo pueda explotar. Dada un función objetivo incomprensible, no hay manera de seleccionar un algoritmo por sobre otro. Si se encuentra uno que muestra un mejor rendimiento que el resto, es fruto de la casualidad [170].

Una rama de investigación más constructiva es aquel que intenta definir el conjunto de funciones donde el NFL no aplica. En [176] se propone el

término de *"searchable"* para clasificar este conjunto de funciones, así como un algoritmo de propósito general que probablemente tiene un rendimiento superior en éste conjunto de funciones que un algoritmo de búsqueda aleatoria (*random search*).

En esta tesis se asume este último enfoque, donde se considera que es posible la construcción de un algoritmo con mejor rendimiento que otros en un limitado subconjuntos de funciones. Si bien no se hace ningún intento para demostrar esto, los trabajos presentados tanto en conjunto de funciones de prueba como en problemas del mundo real confirman este enfoque.

PARTE II

Optimización por Cúmulo de Partículas. Nuevas Variantes

CAPÍTULO 4

Optimización por Cúmulo de Partículas (PSO)

La optimización mediante el cúmulo de partículas PSO (*Particle Swarm Optimization*) es una metaheurística poblacional propuesta por Kennedy y Eberhart [84]. Está basada en la simulación de modelos sociales simples e inspirada en el comportamiento social del vuelo de las bandadas o el movimiento de los bancos de peces.

PSO utiliza lo que se conoce como Inteligencia de Cúmulo (*Swarm Intelligence*) y plantea una "metáfora social" a partir de la cual, los individuos conviven en una sociedad y tienen una opinión que es parte de un "conjunto de creencias" (el espacio de búsqueda) compartido por los demás individuos. De la interacción entre ellos, surge una estrategia que conduce la búsqueda hacia la solución óptima.

El libro de Kennedy y Eberhart [177] describe muchos aspectos filosóficos de PSO y la inteligencia de cúmulo. Poli hizo una cobertura exhaustiva de las aplicaciones de PSO en [178] y [179].

Cada individuo puede modificar su propia opinión basándose en tres factores:

- conocimiento actual (su capacidad para resolver el problema).
- conocimiento histórico o experiencias anteriores (su memoria o conocimiento cognitivo).
- conocimiento de los individuos situados en su vecindario (su conocimiento social).

Es decir, los tres factores descritos anteriormente indican que intervienen aspectos sociales, culturales y de imitación.

Los seres humanos también utilizan la *inteligencia de cúmulo* en la resolución de problemas de optimización. Por ejemplo, en un incendio, ante la imposibilidad de encontrar una vía de escape rápida, los humanos tienden a seguir el comportamiento de la mayoría; por tal motivo, si alguien advierte que todos corren en determinada dirección,

lo natural es que adopte el mismo comportamiento sin saber efectivamente si esa dirección es la correcta. Según este enfoque, el conocimiento actual de cada individuo se ve afectado por el entorno.

En [84], PSO utiliza una población de tamaño fijo. Cada elemento de la población, denominado partícula, es una solución del problema y sus movimientos se encuentran acotados al espacio de búsqueda. Este espacio se encuentra definido de antemano y no se permite que las partículas se desplacen fuera de él.

4.1 PSO Continuo

En su concepción original PSO se planteó para espacios de búsqueda de variables continuas n-dimensionales [84], aunque poco tiempo después los autores originales propusieron una versión para espacios discretos binarios [85]. Esta última comparte los componentes y fundamentos básicos, pero adaptando las fórmulas de cálculo de ubicaciones a espacios binarios.

A continuación es posible apreciar que la formula de actualización de ubicación y de velocidad de las partículas trabajan sobre la representación matemática directa de los puntos que constituyen cada individuo, sin mediar conversiones o transformaciones vinculadas con la representación.

Características

En PSO, cada individuo o partícula está siempre en continuo movimiento explorando el espacio de búsqueda y nunca muere.

Cada partícula está compuesta por tres vectores y dos valores de fitness:

- Vector $Xi=(xi1,xi2,\ldots,xin)$ almacena la posición actual de la partícula en el espacio de búsqueda.
- Vector $pBesti=(pi1,pi2,\ldots,pin)$ almacena la mejor solución encontrada por la partícula hasta el momento.
- Vector velocidad $Vi=(vi1,vi2,\ldots,vin)$ almacena el gradiente (dirección) según el cual se moverá la partícula.
- El valor fitness $fitness\_x_i$ almacena el valor de aptitud de la solución actual (vector X_i).

- El valor fitness $fitness\_pBest_i$ almacena el valor de aptitud de la mejor solución local encontrada hasta el momento (vector $pBest_i$)

La población se inicializa generando las posiciones y las velocidades iniciales de las partículas aleatoriamente. Una vez que la población ha sido creada, los individuos comienzan a moverse por el espacio de búsqueda por medio de un proceso iterativo.

Con la nueva posición del individuo, se calcula y actualiza su fitness ($fitness\_x_i$). Además, si el nuevo fitness del individuo es el mejor encontrado hasta el momento, se actualizan los valores de mejor posición $pBest_i$ y fitness $fitness\_pBest_i$.

Como se explicó anteriormente, el vector velocidad es modificado tomando en cuenta su experiencia y su entorno, según la siguiente expresión:

$$V_{id}(t+1)=w.V_{id}(t)+Õ_1.rand_1.(pBest_{id}-X_{id}(t))+Õ_2.rand_2.(G_{id}-X_{id}(t)) \quad (4.1)$$

donde

- w representa el factor inercia y decrece en forma lineal a medida que avanzan las iteraciones del algoritmo[180].
- Las constantes $Õ_1$ y $Õ_2$ se denominan coeficientes de aceleración ya que permiten modelar la tasa de cambio de cada partícula. Son las encargadas de medir la importancia que se le da a los factores cognitivo y social.
- $rand_1$ y $rand_2$ son valores aleatorios pertenecientes al intervalo (0,1)
- G_i representa la posición de la partícula con el mejor *pBest* en el entorno de X_i (*lBest* o *localbest*) o del todo el cúmulo (*gBest* o *globalbest*).

Los valores de w, $Õ_1$ y $Õ_2$ son importantes para asegurar que el algoritmo converja. Una ventaja de esta técnica es que no requiere ninguna información de gradiente. Esto permite aplicarla a problemas donde la información de gradiente o bien no es accesible o bien es

muy costosa de calcular. Para más detalles sobre la elección de estos valores consultar [181] y [182].

Finalmente, se actualiza la posición de la partícula de la siguiente forma:

$$X_{id}(t+1)=X_{id}(t)+V_{id}(t+1) \qquad (4.2)$$

En la figura 4.1 se detalla una representación gráfica del movimiento de una partícula en el espacio de soluciones.

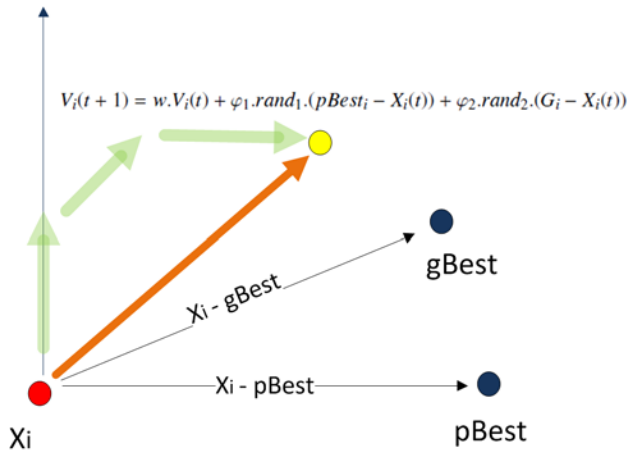


Figura 4.1: Movimiento de una partícula en el espacio de soluciones.

El algoritmo 2 contiene el pseudocódigo del procedimiento de búsqueda de PSO. Es importante notar que PSO es una metaheurística que hace muy pocas suposiciones (por no decir ninguna) con respecto al problema que se busca resolver y se desempeña muy bien en espacios altamente dimensionales.

Como contrapartida debe mencionarse que ninguna metaheurística, PSO incluida, garantiza que la solución óptima sea alcanzada. PSO no utiliza la información de ningún gradiente para realizar la búsqueda, esto implica que la función de aptitud a utilizar no necesita ser diferenciable como es requerido por los métodos de optimización clásicos, como descenso de gradiente y los métodos cuasi-Newton. Esto último permite que PSO sea utilizado en problemas de

optimización cuya función de aptitud sea ruidosa o se encuentre parcialmente definida.

El algoritmo es de implementación sencilla y muestra una convergencia rápida a buenas soluciones. Tanto las posiciones iniciales como las velocidades iniciales se calculan utilizando generadores aleatorios con distribución uniforme en intervalos predefinidos. La versión canónica del algoritmo considera al total de la población como el vecindario de cada partícula, por lo que cada una de ellas puede visualizar la posición de todo el resto. Esta versión es conocida como PSO global, y el intercambio de información entre todos los integrantes del cúmulo es directo.

Existen versiones de este algoritmo donde el concepto de vecindario se restringe a un subconjunto de la población, denominadas PSO local [183]. En estas versiones, se definen diferentes topologías de vecindarios que acotan el intercambio de información entre las partículas, logrando que la información de las mejores ubicaciones se disemine en forma mas lenta. De esta forma, se intenta evitar convergencias prematuras a soluciones sub-óptimas.

Algorithm 2: Algoritmo PSO Básico

```
Pop = CrearPoblacion(N);
while no se alcance la condición de terminación do
    for i = 1 to size(Pop) do
        evaluar partícula x_i del cúmulo Pop;
        if fitness(x_i) es mejor que fitness(pBest_i) then
            pBest_i ← x_i;
            fitness(pBest_i) ← fitness(x_i);
    for i = 1 to size(Pop) do
        elegir g_i de acuerdo al criterio de vecindario usado;
        v_i ← w.v_i + (φ1.rand1.(pBest_i − x_i) + φ2.rand2.(g_i − x_i);
        x_i ← x_i + v_i;
Result: la mejor solución encontrada
```

Una breve descripción de cómo funciona el algoritmo es la siguiente: Inicialmente, se distribuyen partículas en forma aleatoria en el espacio de búsqueda. Luego, alguna partícula se identifica como la mejor de todas (*gBest*) en función de su valor de aptitud (*fitness value*). Todas las partículas del cúmulo son atraídas por esta partícula, y a su vez se mueven también en la dirección de sus propias soluciones mejores, descubiertas hasta el momento. De vez en cuando las partículas sobrepasan la ubicación de su objetivo y exploran el espacio de búsqueda más allá de la mejor partícula actual. Todas las partículas

tienen la oportunidad de conocer una mejor ubicación en el camino y se pueden transformar en la nueva mejor partícula (*gBest*). En este caso las otras partículas cambiarán de dirección y se dirigirán hacia esta nueva ubicación. Asumiendo que el espacio de búsqueda es continuo, es probable que una buena solución esté rodeada de soluciones igualmente buenas o mejores. Al acercarse a la mejor solución actual, desde diferentes direcciones en el espacio de búsqueda, son altas las posibilidades de que las buenas soluciones vecinas sean descubiertas por algunas de las partículas del cúmulo.

Análisis del algoritmo

Muchas interpretaciones se han derivado de la estructura de este algoritmo. Dentro de las corrientes más usuales podemos destacar la que le asigna un comportamiento psico-social al movimiento de las partículas [184]. Considerando el segundo término de la ecuación de velocidad (4.1),

$$\tilde{O}_1.rand_1.(pBest_{id} - X_{id}(t))$$

podemos identificar a este término como el componente cognitivo del movimiento de la partícula, ya que sólo considera su mejor experiencia personal. El tercer término

$$\tilde{O}_2.rand_2.(g_{id} - X_{id}(t))$$

corresponde al componente social, dado que considera la mejor ubicación de la partícula líder del enjambre.

Poco se sabe de la influencia específica de cada uno de estos componentes, pero experimentos han demostrado que el componente social parece tener más influencia en el proceso de búsqueda.

Por lo tanto, la definición de las ubicaciones de las partículas depende tanto de un componente personal cognitivo como de un componente social. Este último es una forma de colaboración entre las partículas [185], dado que se intercambia la información de la mejor ubicación encontrada hasta el momento.

PSO presenta la estructura tradicional de los algoritmos evolutivos. Cuenta con una población de soluciones, tiene además un componente estocástico en su forma de actualización de ubicaciones. La mejor ubicación personal de las partículas actúa como una forma débil de selección. La fórmula de actualización de velocidad es similar al

operador de cruza en los algoritmos genéticos con representación real [186] [187].

La principal diferencia entre PSO y otras técnicas evolutivas radica en que el primero mantiene información tanto de posiciones como de velocidad (cambios en las posiciones). El resto de las técnicas evolutivas en general mantiene información sólo de las ubicaciones.

El movimiento de las partículas en el espacio de búsqueda se ha descrito como un vuelo sobre el espacio n-dimensional [183]. La terminología se debe a que la primera propuesta de esta técnica estaba dirigida a simular en forma visual el comportamiento de una bandada, para realizar una simulación por computador [188]. Estas formaciones aparentan tener un líder general que imprime la dirección de la formación, y no parece verse afectada por la cantidad de integrantes, dado que se han detectado bandadas de varios kilómetros de largo. Si consideramos que la capacidad de procesamiento y de visualización es limitada, podemos inferir que cada ave más que seguir a un líder general, define su movimiento a partir de la información que le llega de sus aves vecinas.

Este comportamiento de las aves tiene importantes ventajas evolutivas: protección de los depredadores, mejora la conservación del pool genético y permite realizar búsqueda de alimentos en áreas extendidas.

El autor en [188] utiliza tres simples reglas para modelar el comportamiento

1. Evitar colisiones
2. Sincronización de velocidad
3. Vuelo centrado en la bandada

Para lograr un comportamiento similar a las aves, se tuvo que considerar la información de los vecinos de cada ave virtual. Este nos rememora al modelo de PSO local (*lBest*). Es interesante notar que en la misma investigación el autor describe que si todas las aves se guían directamente por la posición de un líder, todas las aves tienden a colisionar en el centroide de la bandada. Esto es una descripción bastante exacta de lo que ocurre con la convergencia del modelo de PSO global (*gBest*).

En el algoritmo PSO las partículas carecen de masa, por lo que no colisionan y pueden ocupar la misma posición en forma simultánea. Si bien la denominación más adecuada seria puntos, al considerar

conceptos de velocidad y aceleración, la denominación de partículas parece ser indicada. Asimismo, influyó el nombre de *particle system* que se utiliza en el computación gráfica para renderizar efectos como humo o fuego [188].

Convergencia

Se han realizado cambios a la versión original de PSO (canónica) para mejorar su índice de convergencia. Estos cambios en general involucran modificaciones en las fórmulas de actualización de velocidad, sin variar la estructura básica del algoritmo.

Ponderación de la Inercia

Uno de los cambios mas conocidos es la incorporación de un factor de ponderación (*inertia weight*) a la variable de inercia *w*, propuesto por Shi y Eberhart en [189]. De acuerdo a estos autores, esta variable debe tener valores entre 0.8 y 1.2 para obtener los mejores resultados. Con valores mayores a 1.2 la convergencia general del algoritmo parece disminuir [189].

Una configuración muy usada de este valor de inercia es una reducción lineal considerando como valor inicial 0.9 y como valor final 0.4 [190]. Esta configuración permite recorrer porciones muy extensas del espacio de búsqueda en las primeras iteraciones del algoritmo. A lo largo de la ejecución, al reducir este valor la magnitud del cambio en la posiciones tiende a disminuir, afinando la búsqueda en porciones más reducidas. De esta forma se regula el equilibrio entre la capacidad de exploración del algoritmo (búsqueda de las zonas más prometedoras del espacio del problema) y la capacidad de explotación (búsqueda de soluciones óptimas en un espacio reducido) [191]; equilibrio deseable en todo algoritmo de búsqueda y optimización. Este parámetro se asemeja a la variable temperatura del algoritmo Recocido Simulado.

Factor de Constricción

Clerc ha realizado investigaciones muy interesantes desarrollando el concepto de factor de constricción[192] [193]. Este factor asegura la convergencia del algoritmo, evitando la explosión del cúmulo que ocurre bajo determinadas condiciones. El autor define una

combinación de valores para las variables *w*, $Æ_1$ y $Æ_2$ tal que la convergencia se vea asegurada.

Con la correcta selección de estos valores se evita la necesidad de limitar la velocidad que puede tomar una partícula, practica usual para evitar que el algoritmo se vaya de los rangos de valores normales.

$$v_{id}(t+1)=K[w.v_{id}(t)+Õ_1.rand_1.(pBest_{id}-x_{id}(t))+Õ_2.rand_2.(g_{id}-x_{id}(t))] \quad (4.3)$$

$$K=\frac{2}{|2-\phi-\sqrt{\phi^2-4\phi}|} \quad (4.4)$$

y

$$Õ=c_1+c_2 \ , \ Õ>4$$

Reemplazando y despejando en la fórmula original se llega a la siguiente determinación de valores

$$c1=c2=1.4962 \ ; \ w=0.7298$$

Estos valores de configuración se utilizan cuando se implementa la versión de Clerc con el factor de constricción [194] [195].

Diversidad

En los procesos de búsqueda basados en algoritmos estocásticos, mantener la diversidad de la población mejora las capacidades de exploración del algoritmo y evita la convergencia prematura a óptimos locales.

En forma empírica se ha demostrado la tendencia a la convergencia prematura que muestra la versión global de PSO, redundando en una alta probabilidad de quedar atrapado en soluciones sub-óptimas. Esta fuerte convergencia se origina en que la partícula líder del cúmulo actúa como un fuerte *atractor* de toda la población, generando una convergencia acelerada en esta última.

A los efectos de limitar el efecto de atracción que genera la mejor partícula, se ha trabajado con el concepto de vecindario propuesto en la versión PSO local. Este concepto hace que las partículas no puedan

ver el enjambre completo, sino un subconjunto de la población total. Este mecanismo provoca que el índice de convergencia baje, dado que la información de la mejor ubicación global se expande dentro de la población de una forma más lenta.

Entre las propuestas destacadas puede mencionarse a la implementación de un PSO local por parte de Suganthan [196], que usa el concepto de vecindario espacial, en donde la interacción de cada partícula se realiza con las partículas más próximas en el espacio de búsqueda, utilizando el concepto de distancia euclidea. Los resultados publicados por el autor muestran que el rendimiento es mejor que la versión de PSO global.

Otras propuestas están basadas en el concepto de topología de la población. En este concepto, el vecindario de cada partícula se define en forma estática y se mantiene a lo largo del algoritmo. En [197] se estudió como varía el flujo de información entre las partículas con la utilización de diferentes topologías de la población. Las primeras topologías investigadas por Kennedy son las de anillo (figura 4.2) y las de rueda (figura 4.3), con un hub central. Luego, se intercambian en forma aleatoria las relaciones.

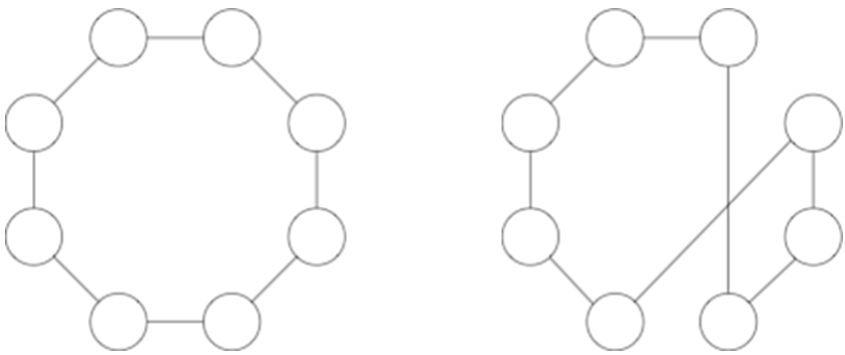


Figura 4.2: Topologías anillo: a) Clásico, b) Aleatorio

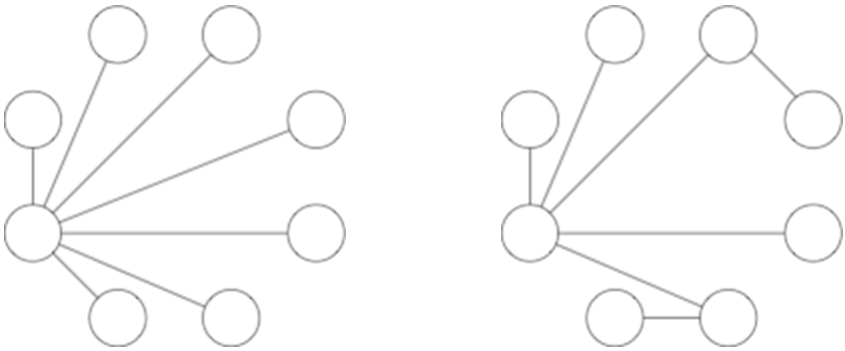


Figura 4.3: Topologías estrella: a) Clásico, b) Aleatorio

Los resultados presentados por el autor sugieren que la utilización del concepto de vecindario mejora las capacidades de búsqueda del algoritmo, reduciendo la posibilidad de convergencia prematura. Además sugiere que la definición de la topología óptima a utilizar es dependiente del problema. Nótese que el concepto de vecindario de este párrafo se estructura en el espacio de índices de las partículas, no en el espacio de búsqueda.

En la propuesta original de PSO las partículas definen su dirección futura basada en su propia experiencia, así como en la posición de la mejor partícula de la población / vecindario. Estudios realizados sobre el comportamiento humano muestran que las personas, más que seguir los pensamientos de un individuo en particular, siguen mas bien un conjunto de creencias formado por determinado grupo. Basado en este concepto, Kennedy propuso una versión donde cada partícula pertenece a un *cluster*, y la ubicación de la mejor partícula (*gbest*) se reemplaza por el centroide de ese *cluster* [198]. De esa forma, se representa mejor el comportamiento humano. Los resultados reportados no fueron demasiados prometedores, y el aumento de la complejidad del algoritmo es considerable por el esfuerzo necesario para calcular los clusters.

Otras técnicas para aumentar la diversidad se basan en el uso de de sub-poblaciones. Las mismas actúan como compartimientos estancos, e intercambian individuos de acuerdo a una cantidad predeterminada de iteraciones. Esta idea fue tomada de otros algoritmos evolutivos [199]. Los resultados reportados en [200] parecen indicar que no existen beneficios importantes por el hecho de utilizar sub-poblaciones. No obstante, se considera que dados los resultados prometedores alcanzados en otros algoritmos evolutivos vale la pena explorar esta alternativa [104].

Además, al incorporar el concepto de sub-población la paralelización del algoritmo se hace naturalmente.
En el siguiente capítulo se realiza una implementación paralela del algoritmo PSO, reportándose los resultados alcanzados tanto en lo referente a su capacidad de optimización como a la medición del *speed up* alcanzado.

4.2 PSO Binario

En diversas aplicaciones prácticas cada vez es más frecuente la presencia de problemas de optimización que involucran variables que deben tomar valores discretos. Se trata de problemas de naturaleza combinatoria donde una de las formas de resolverlos es a través de la discretización de valores continuos. Considerando que cualquier problema de optimización, discreto o continuo, puede ser expresado en notación binaria, Kennedy y Eberthart en [85], consideraron de utilidad la posibilidad de disponer de una versión binaria discreta de PSO para problemas de optimización discretos que denominaron *PSO Binario*.

Características

En un espacio binario, una partícula se mueve en las esquinas de un hipercubo, cambiando los valores de los bits que determinan su posición. En este contexto, la velocidad de una partícula puede ser vista como la cantidad de cambios ocurridos por iteración o la distancia de Hamming entre las posiciones de la partículas en el instante t y el t+1. Una partícula con 0 bits cambiados, de una iteración a la otra, no se mueve mientras que otra partícula que cambia por el valor contrario todos sus bits, se desplaza a velocidad máxima.

Aún no se ha dicho en qué consiste el vector velocidad. Repitiendo la notación del capítulo anterior, la partícula de PSO binario está formada por:

- $X_i=(x_{i1},x_{i2},\ldots,x_{in})$, vector que almacena la posición actual de la partícula en el espacio de búsqueda. Es decir que se trata de un vector binario.
- $pBest_i=(p_{i1},p_{i2},\ldots,p_{in})$, vector que almacena la mejor solución encontrada por la partícula hasta el momento. Por lo tanto, también es binario.
- Vector velocidad $V_i=(v_{i1},v_{i2},\ldots,v_{in})$ es un vector de valores reales.

- *fitness_*x_i, almacena el valor de aptitud de la solución actual (vector X_i).
- *fitness_pBest*$_i$, almacena el valor de aptitud de la mejor solución local encontrada hasta el momento (vector *pBest*$_i$).

En este caso, el vector velocidad está formado por las probabilidades de cambio que tiene cada bit que determina la posición de la partícula de tomar el valor 1. Es decir que una partícula en cada dimensión sólo puede tomar el valor 0 o el valor 1, donde v_{id} representa la probabilidad de que el bit x_{id} tome el valor 1. En otras palabras, si v_{id}=0.35 hay una chance del 35% de que x_{id} se convierta en 1 y una chance de 0.65% de que se convierta en 0. Si la mejor posición de la partícula en la dimensión *d* fuere 0, la expresión ($p_{id}-x_{id}$) podrá dar como resultado −1, 0 o 1 y puede utilizarse para pesar la probabilidad de cambio v_{id} en el próximo paso.

En resumen, el valor de la velocidad para la partícula *i* en la dimensión *d* se actualiza según (4.5)

$$v_{id}(t+1)=w.v_{id}(t)+Õ_1.rand_1.(p_{id}-x_{id}(t))+Õ_2.rand_2.(g_{id}-x_{id}(t)) \quad (4.5)$$

Nótese que si bien las ecuaciones (4.5) y (4.1) coinciden, pero ahora p_{id} y x_{id} son enteros en {0,1} y v_{id} por ser una probabilidad debe ser acotada al intervalo [0,1]. Para lograr esto último se utiliza una función sigmoide como se indica en (4.6)

$$sig(v_{id}(t))=\frac{1}{1+e^{-v_{id}(t)}} \quad (4.6)$$

Luego, el vector posición de la partícula se actualiza de la siguiente forma

$$x_{id}(t+1)=\begin{cases} 1 si\ rand() < sig(v_{id}(t+1)) \\ 0 en\ otro\ caso \end{cases} \quad (4.7)$$

donde *rand*() es un número aleatorio con distribución uniforme en [0,1] distinto para cada dimensión.

El algoritmo 3 contiene el pseudocódigo correspondiente.

Algorithm 3: Algoritmo PSO Binario

```
Pop = CrearPoblacion(N);
while no se alcance la condición de terminación do
    for i = 1 to size(Pop) do
        Evaluar Partícula x_i del cúmulo Pop;
        if fitness(x_i) es mejor que fitness(pBest_i) then
            pBest_i ← x_i;
            fitness(pBest_i) ← fitness(x_i);
    for i = 1 to size(Pop) do
        Elegir g_i de acuerdo al criterio de vecindario usado;
        v_i ← w.v_i + (φ1.rand1.(pBest_i − x_i) + φ2.rand2.(g_i − x_i);
        for d = 1 to n do
            if rand() < sig(v_id) then
                x_id = 1
            else
                x_id = 0
Result: la mejor solución encontrada
```

En la versión continua de PSO el valor de v_{id} también se encontraba acotado a un intervalo cuyos valores eran parámetros del algoritmo.

En el caso discreto, el valor de la velocidad es acotado según (4.8)

$$|v_{id}|<Vmax \qquad (4.8)$$

siendo *Vmax* un parámetro del algoritmo. Es importante notar que *Vmax* debe tomar valores adecuados para permitir que las partículas sigan explorando mínimamente alrededor del óptimo. Por ejemplo, si *Vmax*=6 las probabilidades de cambio $sig(v_{id})$ estarán limitadas a 0.9975 y 0.0025. Esto permite una reducida probabilidad de cambio. Pero si *Vmax* tomara un valor extremo, por ejemplo 50, sería extremadamente poco probable que una partícula cambie su posición luego de alcanzar un óptimo (que tal vez sea local).

Es importante remarcar que la incorporación de la función sigmoide cambia radicalmente la manera de utilizar el vector velocidad para actualizar la posición de la partícula. En PSO continuo, el vector

velocidad toma valores mayores al inicio para facilitar la exploración del espacio de soluciones y al final se reduce para permitir que la partícula se estabilice.

En PSO binario ocurre precisamente todo lo contrario. Los valores extremos, al ser mapeados por la función sigmoide, producen valores de probabilidad similares, cercanos a 0 o a 1, reduciendo la chance de cambio en los valores de la partícula. Por otro lado, valores del vector velocidad cercanos a cero incrementan la probabilidad de cambio. Es importante considerar que si la velocidad de una partícula es el vector nulo, cada uno de los dígitos binarios que determina su posición tiene probabilidad 0.5 de cambiar a 1, siendo esta la situación más aleatoria que puede ocurrir.

Cada partícula incrementa su capacidad exploratoria a medida que el vector velocidad reduce su valor; cuando v_{id} tiende a cero, $\lim_{t'} \cdot sig(v_{id}(t))=0.5$, permitiendo que cada dígito binario tome el valor 1 con probabilidad 0.5. Es decir que puede tomar cualquiera de los dos valores. Por el contrario, cuando los valores del vector velocidad se incrementan, $\lim_{t'} \cdot sig(v_{id}(t))=1$, y por lo tanto todos los bits tienden a 1, mientras que cuando el vector velocidad decrece, tomando valores negativos, $\lim_{t'} \cdot sig(v_{id}(t))=0$ y todos los bits cambian a 0. Es importante remarcar que limitando los valores del vector velocidad entre −3 y 3, $sig(v_{id}) \in [0.0474, 0.9526]$, mientras que para valores superiores a 5, $sig(v_{id}) \simeq 1$ y para valores inferior a −5, $sig(v_{ij}) \simeq 0$. En [177] se recomienda limitar los valores de la velocidad al rango [-4,4]. Esta versión de PSO binario no utiliza ni los factores de constricción ni la ponderación de la inercia dado que fue una versión previa a esos desarrollos.

En [201] se hace una comparación entre un algoritmo genético binario y una versión binaria de PSO, donde se infiere que este último tiene un mejor desempeño y una mejor velocidad de convergencia que el algoritmo genético. Existen también versiones para espacios discretos, donde el tratamiento elegido es discretizar el valor de las variables de los vectores de decisión [202].

En esta tesis se hace una propuesta de algoritmo PSO binario donde se obtuvieron resultados prometedores [203]. Esa propuesta es comparada con el algoritmo PSO binario original y con la versión presentada en [204] dado que es una de las variantes con mejor desempeño reportado.

Variante de PSO Binario

Si bien se han definido distintas alternativas al algoritmo PSO binario descripto en la sección anterior, como por ejemplo [205], [206] y [207], en esta tesis se ha seleccionado el artículo [204] por su buen desempeño y facilidad de implementación.

En la versión de PSO binario definida en [204], se cuestiona la dificultad de selección de parámetros para el algoritmo de Optimización por Cúmulo de Partículas binario. En especial se menciona la importancia que el parámetro de inercia tiene en la capacidad exploratoria del método.

Por este motivo, en [204] la velocidad de una partícula ya no se define como la probabilidad de cambiar a 1, sino como la probabilidad de cambiar desde su estado previo a su valor complementario.

En esta nueva definición, la velocidad de partícula como también sus parámetros tienen el mismo rol que en la versiones de PSO descriptas previamente. La mejor posición que visitó la partícula $pBest_i$ y la mejor posición global *gBest* son actualizadas como en la versión continua o binaria de PSO.

Sin embargo, por cada partícula se introducen dos vectores: $V^{0,i}$ es la probabilidad que los bits de la partícula cambien a cero y $V^{1,i}$ es la probabilidad que los bits de la partícula cambien a uno. Dado que en la ecuación de actualización de estas velocidades el término de inercia es utilizado, estas velocidades no son complementarias. La probabilidad de cambio en el j-ésimo bit de la i-ésima partícula es definido como sigue:

$$v_{ij}^{c} = \begin{cases} v_{ij}^{1} si\ x_{ij}=1 \\ v_{ij}^{0} si\ x_{ij}=0 \end{cases} \qquad (4.9)$$

La manera en que estos vectores son actualizados es la siguiente: Si el j-ésimo bit en la mejor posición global es 0 (cero) o si el j-ésimo bit en la mejor solución hallada por la partícula es 0, la velocidad $v^{0,ij}$ es incrementada y la probabilidad de cambiar a 1 decrece. Por el contrario, si el j-ésimo bit en la mejor posición global es 1 o si el j-ésimo bit en la mejor solución hallada por la partícula es 1, $v^{1,ij}$ es incrementada y $v^{0,ij}$ decrece.

Usando este concepto, se extraen las siguientes reglas:

$$si\ pBest_{ij}=1\ entonces\ d^{1}_{ij,1}=c_1 r_1\ \ and\ d^{0}_{ij,1}=-c_1 r_1$$

$$si\ pBest_{ij}=0\ entonces\ d^{0}_{ij,1}=c_1 r_1\ \ and\ d^{1}_{ij,1}=-c_1 r_1$$

$$si\ gBest_{j}=1\ entonces\ d^{1}_{ij,2}=c_2 r_2\ \ and\ d^{0}_{ij,2}=-c_2 r_2$$

$$si\ gBest_{j}=0\ entonces\ d^{0}_{ij,2}=c_2 r_2\ \ and\ d^{1}_{ij,2}=-c_2 r_2$$

donde $d^{1,ij}$ y $d^{0,ij}$ son dos valores temporarios, r_1 y r_2 son dos valores aleatorios con distribución uniforme en (0,1) que se actualizan en cada iteración. Las constantes c_1 y c_2 son parámetros del algoritmo y se definen a priori.

Luego, los vectores velocidad se actualizan según (4.10) y (4.11)

$$v^{1}_{ij}=wv^{1}_{ij}+d^{1}_{ij,1}+d^{1}_{ij,2} \qquad (4.10)$$

$$v^{0}_{ij}=wv^{0}_{ij}+d^{0}_{ij,1}+d^{0}_{ij,2} \qquad (4.11)$$

donde w es un término de inercia. Desde este enfoque la dirección de cambio, hacia 1 o hacia 0, para cada bit, es considerada por separado. Luego de actualizar la velocidad de las partículas, se obtiene la velocidad de cambio, como se indicó en (4.9).

Finalmente, para obtener la nueva posición de la partícula, se utiliza la función sigmoide definida en (4.6) y se calcula la nueva posición siguiendo lo indicado en (4.12)

$$x_{ij}(t+1)=\begin{cases} \overline{x_{ij}}\ (t), si\ r_{ij}<sig(v_{ij}(t+1)) \\ x_{ij}(t), en\ otro\ caso \end{cases} \qquad (4.12)$$

donde $\overline{x_{ij}}$ es el complemento a 2 de x_{ij}. Es decir, si $x_{ij}=0$ entonces $\overline{x_{ij}}=1$ y si $x_{ij}=1$ entonces $\overline{x_{ij}}=0$. De la misma forma que en el

PSO binario convencional, r_{ij} es un valor aleatorio con distribución uniforme entre 0 y 1.

Propuesta de Implementación de PSO Paralelo

5.1 Introducción

El objetivo de este trabajo consiste en efectuar una comparación entre la versión canónica del algoritmo Optimización por Cúmulo de Partículas (PSO - *Particle Swarm Optimization*) [Ken95] y una versión descentralizada del mismo, basada en islas de poblaciones independientes con migración de partículas entre poblaciones en intervalos predefinidos.
Los problemas de prueba seleccionados como criterio de evaluación corresponden a problemas de optimización numérica con variables reales sin restricciones, tomados del documento **Problem Definitions and Evaluation Criteria for de CEC 2005 Special Session on Real-Parameter Optimization** [208].
Se presenta un informe detallado del rendimiento de ambos algoritmos en las 25 funciones de prueba definidas. Adicionalmente se efectúan dos implementaciones de la versión descentralizada (distribuida), una variante del tipo secuencial a ejecutarse en un único procesador y otra variante paralela a ejecutarse en *m* procesadores en simultáneo, un procesador por cada isla, midiéndose el rendimiento de la versión paralela en una red de maquinas homogéneas.

5.2 Algoritmo propuesto (dPSO)

Se propone el siguiente diseño de algoritmo PSO distribuido. El mismo se corresponde al modelo dEA (*distributed evolutionary algorithms*) [104].

La propuesta consiste en un conjunto de *m* poblaciones, cada uno con una población de *n* partículas, que efectúan escasos intercambios de individuos (partículas). Cada población comienza su procesamiento en forma independiente y cada cierta cantidad *o* de iteraciones intercambian individuos (partículas). Se selecciona la mejor partícula de cada población (*gBest*) y se envía a una población vecina. Esta

partícula remplaza a una partícula aleatoria de la población receptora. Las poblaciones están dispuestas en una topología de anillo estática, por lo que cada población envía su mejor partícula siempre al mismo vecino, y recibe partículas siempre desde otro vecino predefinido (figura 5.1).

Es importante destacar que esta versión del algoritmo es cualitativamente disímil a la versión canónica. El comportamiento de ambos es diferente dado que en la versión distribuida las diferentes poblaciones comparten información durante la ejecución del mismo. Por lo tanto, los resultados alcanzados por esta versión con *m* poblaciones no son estadísticamente similares a la ejecución de *m* corridas del algoritmo canónico.

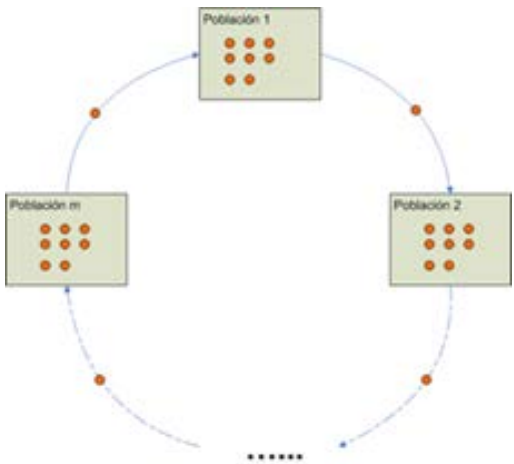


Figura 5.1: dPSO. Topología de anillo estática

Se proponen a continuación dos implementaciones de este algoritmo, una secuencial diseñada para ejecutarse en un solo procesador y una paralela a ejecutarse en más de un procesador (figura 5.2).

Figura 5.2: Dos implementaciones del mismo algoritmo. Implementación secuencial de dPSO (sdPSO)

En el algoritmo 4 se detalla el pseudo-código de la implementación secuencial del algoritmo PSO distribuido.

Esta versión secuencial se corresponde con el modelo distribuido detallado anteriormente, y propone una implementación secuencial a ejecutarse en un solo procesador.

Como se desprende del pseudo-código, en esta versión cada una de las *m* poblaciones efectúa una cantidad máxima de iteraciones (*cant_maxima_de_iteraciones*). Es decir, que si el algoritmo distribuido se ejecuta para *m*=2 poblaciones, la cantidad total de iteraciones realizada por el algoritmo es *m**(*cant_maxima_de_iteraciones*).

El parámetro *o* define el intervalo de intercambio de partículas, es decir cuántas iteraciones realiza el algoritmo antes de enviar/recibir una partícula hacia/desde sus vecinos. El mismo está expresado en iteraciones del algoritmo.

Algorithm 4: Algoritmo PSO con *m* Poblaciones. Implementación Secuencial

```
Inicializar parámetros;
for i = 1 to m do
    Inicializar posición de partículas y velocidad en forma aleatoria;
    Evaluar fitness;
    Calcular gBest;
    for i = 1 to cant_maxima_de_iteraciones do
        for j = 1 to m do
            for k = 1 to o do
                for l = 1 to cant de particulas do
                    Calcular vector velocidad;
                    Calcular nueva posición;
                    Evaluar fitness;
    Calcular gBest y pBest;
    Guardar gBest(m) (Operación enviar);
    for j = 1 to m do
        Recuperar gBest(m − 1) (Operación recibir);
        Reemplazar una partícula por gBest(m − 1) ;
Reportar resultados
```

Por ejemplo, para los siguientes parámetros:

- Cantidad de poblaciones: m=2
- Cantidad máxima de iteraciones: (*cant_maxima_de_iteraciones*) = 15000
- Intervalo de intercambio de individuos: o=50

La cantidad máxima de iteraciones corresponde a cada población, por lo tanto el total de iteraciones para la corrida del algoritmo es igual a 2*15000=30000.

5.3 Implantación paralela de dPSO (pdPSO)

Se presenta a continuación la implementación paralela del algoritmo dPSO (PSO distribuido). Se requieren por lo menos m instancias del algoritmo en ejecución, siendo m>1. En el algoritmo 5 se detalla el pseudo-código de este algoritmo
Nótese que en esta versión cada una de las m poblaciones se ejecuta en un procesador (instancia del algoritmo) diferente. Los procesos de

comunicación donde se realiza el intercambio de partículas entre poblaciones son de naturaleza sincrónica.

5.4 Plataforma Utilizada

Plataforma de Software

Los algoritmos descritos se implementaron en lenguaje de programación Java, utilizando la interface sockets para el pasaje de información entre las distintas instancias del mismo. La modalidad de intercambio de información es sincrónica.

La utilización de Java es muy conveniente dada su facilidad de programación, su orientación a objetos, su portabilidad, y que sólo requiere la instalación de *Java Virtual Machine* para su ejecución.

Algorithm 5: Algoritmo PSO con *m* Poblaciones. Implementación Paralela

```
Inicializar parámetros;
Establecer conexión con Vecinos;
Inicializar posición de partículas y velocidad en forma aleatoria;
Evaluar fitness;
Calcular gBest;
for i=1 to cant_maxima_de_iteraciones do
    for j=1 to o do
        for k=1 to cant_de_partículas do
            Calcular vector velocidad;
            Calcular nueva posición;
            Evaluar fitness;
        Calcular gBest y pBest;
    Enviar mejor partícula;
    Recibir partícula;
    Reemplazar partícula;
Reportar resultados;
```

La utilización de sockets TCP permite un eficiente intercambio de información entre las diferentes PCs de una LAN estándar.

Como entorno de desarrollo se utilizó IDE Eclipse versión 3.1.2 Build id M20060118-1600.

Como entorno de ejecución se utilizó Java Virtual Machine java version $1.5.0_1 1$ Java(TM) 2 Runtime Environment, Standard Edition (build $1.5.0_1 1-b03$) Java HotSpot(TM) Client VM (build $1.5.0_1 1-b03$, mixed mode, sharing) corriendo en estaciones de trabajo con Windows XP Professional versión 5.1.2600 Service Pack 2 Compilacion 2600.

Plataforma de Hardware

La versión secuencial y paralela del algoritmo se ejecutó sobre una red de 4 PCs con las siguientes características, procesador Intel Pentium 2.66 GHz con 512 MB de RAM. Las estaciones de trabajo están conectadas entre sí utilizando una red Ethernet de 100Mb. Como se mencionó anteriormente, el SO es Windows XP Professional.

5.5 Resultados

Como medida de desempeño de los algoritmos se tomaron las 25 funciones de evaluación según [208]. En el mismo se establecieron un conjunto de funciones y parámetros de evaluación para optimización numérica con variables reales sin restricciones.

De acuerdo a lo sugerido en el trabajo de referencia se evaluaron los algoritmos en 10, 30 y 50 dimensiones. Se generó la información siguiendo el protocolo especificado en el documento.

En el anexo *Rendimientos de algoritmos cPSO y dPSO* al final del documento se adjuntan los cuadros correspondientes a la resolución de las 25 funciones de prueba por parte de ambos algoritmos, en 10, 30 y 50 dimensiones, con los correspondientes gráficos de convergencia para 30 dimensiones. Se muestra además un cuadro con el cálculo de la complejidad del algoritmo, así como el detalle de todos los parámetros utilizados en la implementación.

Las figuras 5.3, 5.4, 5.5, 5.6 y 5.7 muestran la convergencia para las distintas funciones de prueba en 30 dimensiones, donde es posible visualizar el comportamiento de ambas versiones del algoritmo.

Parámetros de Ejecución

Parámetros comunes cPSO y dPSO

Cantidad máxima de evaluación de funciones
10 D = 100000
30 D = 300000
50 D = 500000
Factor inercia (w) con reducción lineal entre 0.9 y 0.4
Factor cognitivo Alfa1 (a1) = 2
Factor social Alfa2 (a2) = 2
Rango de velocidad = [-4,4]
Acción a tomar cuando se exceden límites del dominio = Saturación en valor límite

Parámetros diferentes entre cPSO y dPSO

Tamaño de la población
cPSO = 20
dPSO = 5
Cantidad de Poblaciones
cPSO = 1
dPSO = 4
Intervalo de intercambio de partículas
cPSO = No aplica
dPSO = 50

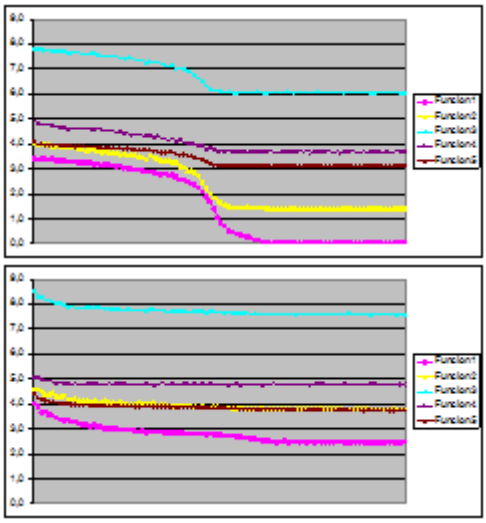


Figura 5.3: Gráfico de Convergencia cPSO (izquierda) y dPSO (derecha). Funciones 1 a 5 en 30 dimensiones

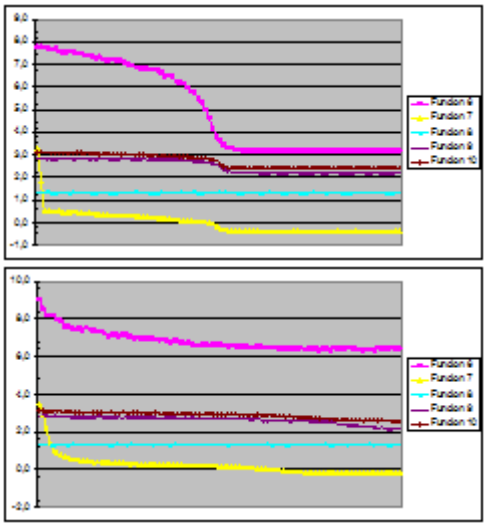


Figura 5.4: Gráfico de Convergencia cPSO (izquierda) y dPSO (derecha). Funciones 6 a 10 en 30 dimensiones

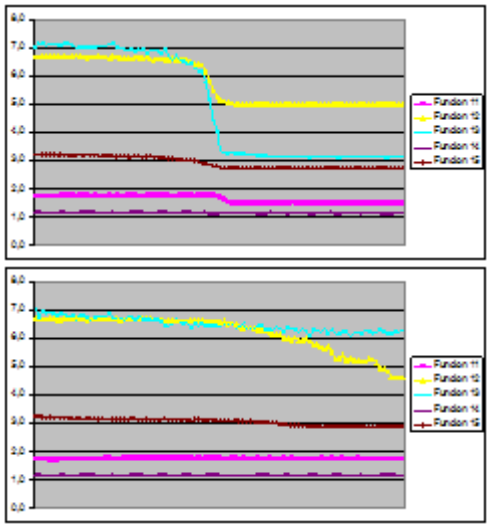


Figura 5.5: Gráfico de Convergencia cPSO (izquierda) y dPSO (derecha). Funciones 11 a 15 en 30 dimensiones

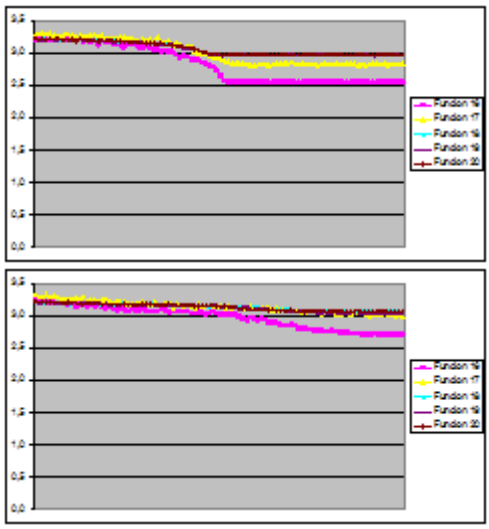


Figura 5.6: Gráfico de Convergencia cPSO (izquierda) y dPSO (derecha). Funciones 16 a 20 en 30 dimensiones

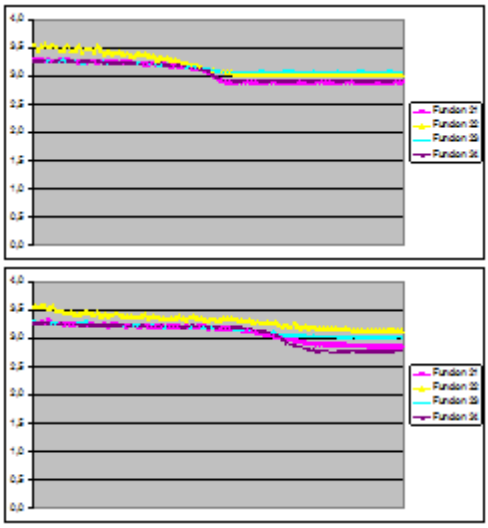


Figura 5.7: Gráfico de Convergencia cPSO (izquierda) y dPSO (derecha). Funciones 21 a 24 en 30 dimensiones

Se comparó el rendimiento de ambos algoritmos realizando un análisis de varianza. Se tomaron las corridas a comparar y se efectúo la prueba de Kolmogorov-Smirnov para determinar si las muestras tienen una distribución normal. En aquellos casos que los conjuntos de muestras presentaron una distribución normal, se calculo el estadístico t de Student. Para aquellas muestras que no presentaron una distribución normal, se efectúo la prueba no paramétrica de Kruskal-Wallis [118] [115] [119]. En todos los casos, si el p-value es menor que 0.05, se considera que las diferencias son estadísticamente significantes.

Del estudio de los resultados se desprende que el algoritmo cPSO superó en rendimiento al dPSO en la mayoría de los problemas de prueba utilizados. En la tabla 5.2 se muestra un resumen del desempeño de los algoritmos, indicando la cantidad de problemas donde cada algoritmo se mostró superior en las diferentes dimensiones evaluadas.

Dimensiones	10	30	50
cPSO	10	13	15
dPSO	1	5	3
S/D	14	7	7
	25	25	25

Tabla 5.1: Resumen del desempeño de los algoritmos por dimensiones

En la tabla 5.2, se adjunta un detalle identificando el algoritmo que mostró mejor desempeño en cada problema/dimensión. En aquellos casos que no existen diferencias estadísticamente significantes se clasificaron con la sigla SD.

ProbDimen-siones	10	30	50
1	SD	SD	cPSO
2	cPSO	cPSO	cPSO
3	cPSO	cPSO	cPSO
4	SD	cPSO	cPSO
5	SD	cPSO	cPSO
6	SD	SD	cPSO
7	SD	SD	cPSO
8	cPSO	cPSO	cPSO
9	dPSO	dPSO	dPSO
10	SD	SD	SD
11	cPSO	cPSO	cPSO
12	SD	dPSO	dPSO
13	SD	cPSO	cPSO
14	cPSO	cPSO	cPSO
15	SD	SD	SD
16	SD	cPSO	SD
17	cPSO	cPSO	cPSO
18	cPSO	cPSO	cPSO
19	cPSO	cPSO	cPSO
20	cPSO	SD	cPSO
21	SD	dPSO	SD
22	SD	cPSO	SD
23	SD	dPSO	SD
24	SD	SD	SD
25	cPSO	dPSO	dPSO

Tabla 5.2: Detalle del desempeño de los algoritmos por funciones y dimensiones

Rendimiento de la versión paralela

En esta parte del trabajo se efectúa un análisis y un estudio empírico acerca de la conveniencia de realizar una implementación paralela del algoritmo PSO descentralizado (dPSO).
A lo largo del documento se hace referencia al concepto de *speed-up* del algoritmo, sabiendo que la definición de *speed-up* requiere que el algoritmo secuencial contra el cual se compara sea el mejor algoritmo conocido para resolver el problema. Esta última condición no se puede asegurar en este trabajo. Además, en la evaluación realizada se considera una cantidad fija de evaluaciones de la función de fitness, sin tener en cuenta la calidad de los resultados obtenidos. Por lo tanto, tomando la taxonomía definida por Alba y Tomassini [209], cada vez que hablamos de *speed-up* en este trabajo hacemos referencia a una definición débil de *speed-up* con un esfuerzo pre-definido (*weak speed-up with predefined effort*). De esta forma, podemos hacer foco en las mejoras en tiempo obtenidas comparando la versión secuencial (sdPSO) y paralela (pdPSO) del mismo algoritmo.

Análisis de rendimiento

En este análisis se utilizó la función número 13 del reporte **Problem Definitions and Evaluation Criteria for de CEC 2005 Special Session on Real-Parameter Optimization** [208] en 50 dimensiones. Los parámetros del algoritmo se mantuvieron constantes. Como primera medida, se tomaron las mediciones indicadas en la tabla 5.3 con el objetivo de analizar la conveniencia de la paralelización del algoritmo PSO distribuido (dPSO). Todos los tiempos están expresados en microsegundos. Estas mediciones corresponden al tiempo promedio de 1000 ejecuciones.

Latencia de Red	3.554426 E+01
Tiempo promedio de 1 iteración del algoritmo	6.870891 E+04
Tiempo promedio de envío/recepción de 1 partícula	2.030928 E+05

Tabla 5.3: Tiempos de iteración y comunicación

Teniendo en cuenta el tiempo promedio de una iteración del algoritmo, y el tiempo promedio para enviar y recibir una partícula, se calcula el costo de envío de una partícula medido en cantidad de iteraciones, dividiendo el segundo por el primero (tabla 5.5).

Costo de comunicación en iteraciones	2.956

Tabla 5.4: Relación tiempo de comunicación/iteraciones

Sabiendo que se intercambian partículas cada **o** iteraciones, podemos calcular la cantidad de comunicaciones necesarias dividiendo la cantidad máxima de iteraciones por o. Dado que la secuencia de ejecución del algoritmo es

```
o iteraciones
envía/recibe partículas
o iteraciones
envía/recibe partículas
...
o iteraciones
envía/recibe partículas
```

es importante destacar que no tiene sentido realizar el último envío/recepción de partículas, dado que el mismo no se utiliza para nada. Por lo tanto, la cantidad de comunicaciones queda definida como

```
Cant de comunicaciones = (Cant máxima de
iteraciones / o) - 1
```

Tomando como base de cálculo que se realizarán 15000 iteraciones por población, se calcula

```
    Tiempo de ejecución de iteraciones = cant
máxima de iteraciones
           * Tiempo promedio de 1 iteración del
algoritmo
```

Luego, se calcula

```
    Tiempo de comunicación = Cant de
comunicaciones
           *  Tiempo promedio de
      envío/recepción 1 partícula
```

De aquí se desprende en forma directa que el tiempo total de ejecución del algoritmo realizando 15000 iteraciones y 299 eventos de comunicación se puede estimar como

```
Tiempo total estimado = Tiempo de ejecución de
iteraciones
                      + Tiempo de comunicación
```

o ⟶ Cantidad de iteraciones entre intercambios	50
Cantidad máxima de iteraciones ⟶ iteraciones x población	15000
Cant.de comunicaciones	299
Tiempo de ejecución iteraciones	1,030634 E+09
Tiempo de comunicación	6.072475 E+07
Tiempo total estimado	1.091358 E+09

Tabla 5.5: Estimación de tiempo total

Los resultados de estas mediciones se muestran en la tabla 5.5.

En la implementación secuencial (sdPSO) el tiempo de envío/recepción de partículas es despreciable (= 0), dado que en realidad lo que se hace es guardar (enviar) la mejor partícula en un vector y recuperarla (recibir) desde ese mismo vector.

Por lo tanto, se calcula el *speed-up* estimado calculando la razón entre el tiempo de ejecución de las iteraciones y el tiempo total estimado del algoritmo.

```
    Speed-up Estimado = Tiempo de ejecución de
iteraciones
                    / Tiempo total estimado
```

o lo que es lo mismo

```
    Speed-up Estimado = Tiempo de ejecución de
iteraciones /
                     (Tiempo de ejecución de
iteraciones +
                     Tiempo de comunicación)
```

Los resultados obtenidos se muestran en la tabla 5.6.

Speed-Up estimado	0.944

Tabla 5.6: Estimación de speed-up

Es importante destacar que este *speed-up* estimado es el máximo posible a alcanzar en la implementación paralela. Seguramente existirá un tiempo adicional a incurrir en la versión paralela relacionada con la sincronización de los procesos al momento de intercambio de información.

La tabla 5.7 muestra los tiempos reales de ejecución de la versión distribuida del algoritmo PSO (dPSO) con dos poblaciones, tanto en su versión secuencial (sdPSO) como en su versión paralela (pdPSO). Este último ejecutando en red homogénea de dos nodos.

Cantidad de Poblaciones (instancias)	2
Tiempo total de ejecución de sdPSO	1.988993 E+09
Tiempo total de ejecución de pdPSO	1.059673 E+09
Speed-up Real	**0.938**

Tabla 5.7: Tiempos de ejecución de dos islas, en uno y dos procesadores

La tabla 5.8 muestra los tiempos reales de ejecución de la versión distribuida del algoritmo PSO (dPSO) con cuatro poblaciones, tanto en su versión secuencial (sdPSO) como en su versión paralela (pdPSO), este último ejecutando en una red homogénea de cuatro nodos.

Cantidad de Poblaciones (instancias)	4
Tiempo total de ejecución de sdPSO	3.994460 E+09
Tiempo total de ejecución de pdPSO	1.067027 E+09
Speed-up Real	**0.936**

Tabla 5.8: Tiempos de ejecución de 4 islas, en uno y cuatro procesadores

Las mediciones obtenidas son consistentes con el análisis realizado. El costo en tiempo relacionado con la sincronización de los procesos no se muestra significativo.

5.6 Conclusiones

Basado en la versión canónica del algoritmo PSO (cPSO) se propuso un algoritmo PSO distribuido (dPSO). Este algoritmo está formado por islas con poblaciones independientes, con un intercambio de información en forma regular y sincrónica.

Se evalúo el rendimiento de estos dos algoritmos (cPSO y dPSO) utilizando las funciones de prueba para optimización numérica con variables reales definidas en **Problem Definitions and Evaluation Criteria for de CEC 2005 Special Session on Real-Parameter Optimization** [208]. Se generó la información de resultados de acuerdo al protocolo definido.

Del análisis de los resultados se puede concluir que la versión PSO canónica (cPSO) mostró un rendimiento superior que la versión distribuida (dPSO) en la mayoría de los problemas de prueba, considerando el conjunto de parámetros utilizado. Estos resultados son consistentes con [200].

Se realizó además la implementación de la versión distribuida en su variante secuencial (sdPSO) y paralela (pdPSO).

A partir del estudio de tiempos necesarios para cada iteración del algoritmo y para el traspaso de información entre procesos se efectúo un análisis para estimar el índice de *speed-up* de la versión paralela. Se registraron los tiempos reales de ejecución, siendo los mismos consistentes con el análisis efectuado.

Del análisis de los resultados se puede concluir que la paralelización del algoritmo dPSO (PSO distribuido) se muestra como conveniente respecto de los índices de *speed-up*, aunque el rendimiento como optimizador global se ve reducido. Se debe tener en consideración que la versión paralela presentada (pdPSO) requiere de una red homogénea de computadoras.

PSO con Detector de Oscilación

6.1 Introducción

PSO es una técnica de optimización que ha mostrado ser simple de implementar, estable, escalable y con la cual se han obtenido buenos resultados en optimización de funciones, incluso cuando se ha comparado con otras técnicas más difundidas como algoritmos genéticos. No obstante, esta técnica no está exenta de problemas, ya que tiene cierta tendencia a quedar atrapada en mínimos locales, y la estrategia de exploración que utiliza alrededor de mínimos locales se muestra ineficiente, dado que tiende a oscilar en exceso alrededor de los mismos, utilizando muchas iteraciones del algoritmo para alcanzarlos [210].

Teniendo en cuenta este último punto, en la versión del algoritmo presentada en esta investigación se cambia la estrategia de acercamiento a los mínimos locales, cuando se detecta una oscilación alrededor de ellos.

Para lograr este objetivo, se utiliza un procedimiento de búsqueda determinista que minimiza la cantidad de pasos para alcanzar los puntos óptimos. De esta forma, es posible acelerar la velocidad de convergencia del algoritmo, sin aumentar la probabilidad de caer en óptimos locales, siempre y cuando este procedimiento de búsqueda local se utilice en los momentos adecuados de la búsqueda.

6.2 Detector de Oscilación

Como ya se explicó, la nueva versión del algoritmo planteada en esta investigación apunta a hacer más eficiente la estrategia de acercamiento cuando se detecta que una partícula está oscilando alrededor de un mínimo (máximo) local (global). Por lo tanto, el primer paso que se debe realizar es descubrir cuándo una partícula se encuentra en este estado (estado de oscilación). Para ello, la nueva versión del algoritmo analiza el signo de la velocidad resultante en cada iteración, para cada partícula en cada dimensión. Cuando se percibe que existe una secuencia de signos de velocidad contrarios en

una sucesión de iteraciones, se define que la partícula, para esa dimensión en particular, se encuentra en estado de oscilación. Es decir, se asume que si el resultado del componente de velocidad para una partícula y para una dimensión específica cambia de signo en forma alternada en iteraciones sucesivas, es probable que se encuentre oscilando alrededor de un mínimo (máximo) local (global). (Ver figura 6.1).

Una vez detectado este estado, para esta partícula y esa dimensión en particular, la velocidad se actualizará utilizando el procedimiento de búsqueda local detallado en el párrafo siguiente.

Uno de los problemas que surgieron al momento de implementar y probar el proceso de detección de oscilación está relacionado con la definición acerca de cuál es la cantidad de iteraciones sucesivas a considerar. De acuerdo a pruebas empíricas, la cantidad de oscilaciones a considerar se mostró dependiente del problema a optimizar. Por lo tanto, a los efectos de encontrar una solución genérica, se decidió tomar dos valores que mostraron buenos resultados en diferentes tipos de problemas. Luego, se inicializan el 50% de las partículas con un valor, y el 50% con el valor restante. Los dos valores utilizados corresponden a: Sensibilidad Alta = 3 y Sensibilidad Baja = 8.

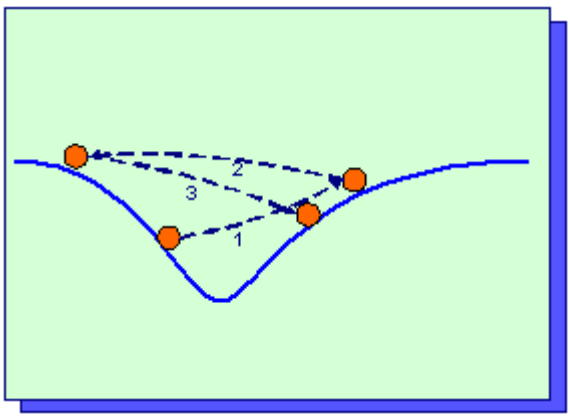


Figura 6.1: Ejemplo de oscilación de una partícula

Luego, con el avance de las corridas del algoritmo, se efectúa un proceso de competencia entre partículas utilizando la técnica de torneo binario. De esta forma, se termina imponiendo para la mayoría de las partículas el parámetro que mejor resultado muestra para la función

que está siendo optimizada. En el párrafo Adaptación de Parámetros se explica este proceso de competencia en detalle.

6.3 Procedimiento de Búsqueda Local

Una vez determinado que la partícula especifica, para una dimensión específica, se encuentra en estado de oscilación, se cambia la fórmula tradicional de la metaheurística PSO de cálculo de velocidad por un Procedimiento de Búsqueda Local Determinista (PBLD). Este procedimiento busca las dos mejores ubicaciones (ubicaciones con mejor *fitness*) dentro de las ubicaciones memorizadas por esa partícula, con signos de velocidad opuestos en la dimensión a considerar, y calcula el punto medio entre estas dos ubicaciones. Este procedimiento se repite en las sucesivas iteraciones, hasta encontrar una ubicación con fitness peor que el anterior.

Como se desprende del funcionamiento del procedimiento de búsqueda local, es necesario guardar para cada partícula la historia de las últimas *n* ubicaciones. En cada iteración, se guarda la ubicación anterior en la memoria, y se elimina la ubicación más antigua. La figura 6.2 ilustra el funcionamiento del Procedimiento de Búsqueda Local.

Es importante destacar que este procedimiento se ejecuta en forma independiente por cada dimensión. Considerando que el Detector de Oscilación es además independiente por cada dimensión, en una iteración simple del mismo algoritmo se da el caso que la velocidad de algunas de las dimensiones de una partícula se actualicen según la formula estándar del algoritmo PSO, y otras dimensiones de la misma partícula en la misma iteración se actualice según este Procedimiento de Búsqueda Local Determinista.

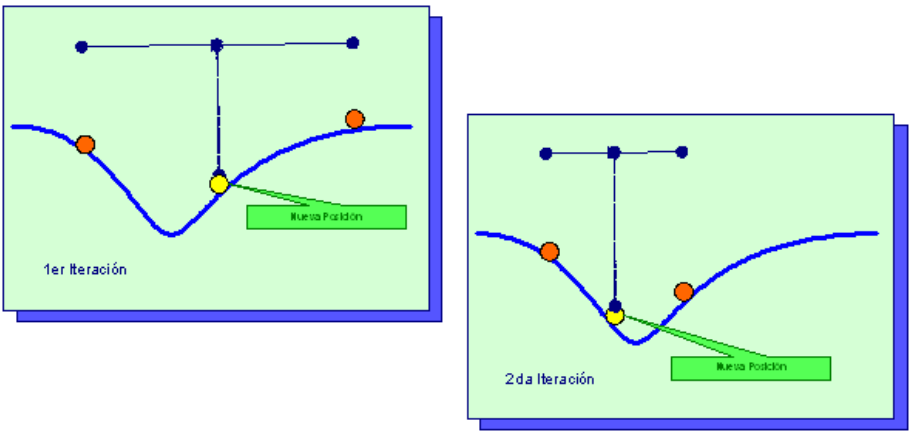


Figura 6.2: Procedimiento de Búsqueda Local

En la versión canónica de PSO, cada vector *n*-dimensional representa una solución potencial completa. Dado que cada paso de actualización se realiza en todas las dimensiones de este vector, es probable que algunos componentes del vector se acerquen al óptimo, y que otros componentes se alejen. Si los componentes que se acercan contribuyen a la función de fitness en mayor medida que los componentes que se alejan, la nueva ubicación completa es considerada mejor que la anterior. Por lo tanto, es importante entender que es posible que el mejoramiento o empeoramiento del valor de fitness no necesariamente corresponda con la conveniencia o no de utilizar este procedimiento de búsqueda local en una dimensión específica. Este problema de inter-dependencia entre las diferentes dimensiones ya existe en el algoritmo PSO tradicional [211]. En este trabajo se asume que si se encuentra una nueva posición con mejor (peor) fitness, es más probable que cada componente del vector *n*-dimensional se acerque (aleje) de la solución óptima o que los beneficios que aportan las dimensiones que se acercan compensan los potenciales perjuicios de las dimensiones que se alejan. Este supuesto esta dado en forma implícita en el planteamiento original de la metaheurística PSO. Existen versiones donde se evalúa el fitness de la partícula luego de actualizar cada dimensión. En este caso, se elimina el efecto descrito a un costo de incrementar notoriamente la cantidad de evaluación de funciones [211].

Uno de los problemas que surgieron al momento de implementar y probar el proceso de búsqueda local está relacionado con la definición

acerca de cuál es la cantidad de memoria a considerar al momento de buscar las dos mejores ubicaciones con signos de velocidad contrarios. De acuerdo a pruebas empíricas, la cantidad de memoria a considerar se mostró dependiente del problema a optimizar. Por lo tanto, a los efectos de encontrar una solución genérica, se decidió tomar dos valores que mostraron buenos resultados en diferentes tipos de problemas. La mitad de las partículas se configuran con uno de los valores que corresponden a Mucha Memoria (se almacenan las últimas 20 posiciones), y la otra mitad se configura con el otro valor que corresponde a Poca Memoria (se almacenan las últimas 8 posiciones). También se encontró en forma empírica que para aquellas funciones cuando el Detector de Oscilación funcionaba mejor con Sensibilidad Alta, convenía utilizar una configuración del Procedimiento de Búsqueda Local Determinista con Mucha Memoria, y para aquellos problemas cuando el Detector de Oscilacion funcionaba mejor con Sensibilidad Baja, es más eficiente utilizar una configuración del Procedimiento de Búsqueda Local Determinista con Poca Memoria. En estudios posteriores se estudiará la relación entre estos dos parámetros del algoritmo propuesto.

6.4 Adaptación de Parámetros

De acuerdo a lo expresado en los dos párrafos precedentes, se encontraron dos configuraciones posibles para el Detector de Oscilación y dos configuraciones posibles para el Procedimiento de Búsqueda Local Determinista (PBLD). Asimismo, se encontró una relación entre estas configuraciones. Por lo tanto, se decidió que al comienzo del algoritmo se definen dos juegos de parámetros para el Detector de Oscilación (sensibilidad) y el Procedimiento de Búsqueda Local Determinista (memoria).

1. Mucha Sensibilidad = 3 ; Mucha Memoria = 20
2. Poca Sensibilidad = 8 ; Poca Memoria = 8

El 50% de las partículas comienzan con la configuración 1), y el 50% de las partículas comienzan con la configuración 2).

Asociado a cada partícula se almacenan dos variables. Las mismas se utilizan para calcular la tasa de éxito del Procedimiento de Búsqueda Local Determinista (PBLD). En la primera variable se registra la cantidad de veces que se activó el procedimiento de búsqueda local PBLD. En la segunda variable, se registra la cantidad de veces que el

procedimiento de búsqueda local PBLD encontró una ubicación con mejor fitness, En esta segunda variable, se va sumando la cantidad en forma acumulada. Se denominará $v1$ a la primera variable y $v2$ a la segunda variable.

Al momento de activarse por primera vez el PBLD

$$v1=v1+1=1.$$

En la siguiente iteración, si el procedimiento encontró una mejor ubicación

$$v2=v2+1=1.$$

En la siguiente iteración, si el procedimiento encontró una mejor ubicación

$$v2=v2+2=3.$$

Por lo tanto, cada nueva iteración que colabora a encontrar una ubicación exitosa, tiene una ponderación mayor. Luego, se define la tasa de éxito como

$$v2/v1.$$

En cada iteración, con una probabilidad de 0.01 se seleccionan dos partículas al azar. Se efectúa un torneo binario, y aquella con mejor tasa de éxito traspasa los parámetros de configuración del Detector de Oscilación (sensibilidad) y el Procedimiento de Búsqueda Local Determinista (memoria) a la partícula con peor fitness. Nótese que en este torneo no se considera el fitness de la partícula, si no solamente se intenta mensurar la colaboración del Procedimiento de Búsqueda Local Determinista al proceso de optimización.

6.5 Algoritmo Propuesto (ocsPSO)

A continuación se detalla el pseudocodigo del algoritmo propuesto. El mismo agrega a la versión original del algoritmo el Detector de Oscilación, el Procedimiento de Búsqueda Local Determinista y el proceso de adaptación de parámetros del Detector de Oscilación (sensibilidad) y del Procedimiento de Búsqueda Local Determinista (memoria)

Algorithm 6: Algoritmo oscPSO propuesto

```
Inicializar parámetros;
Inicializar posición de partículas y velocidad en forma aleatoria;
Evaluar fitness;
Calcular gBest ;
for i=1 to cant maxima de iteraciones do
    for j=1 to cant de particulas do
        for n=1 to cant de dimensiones do
            if modo_oscilacion = VERDADERO then
                Calcular vector velocidad con Procedimiento de Busqueda Local;
            else
                Calcular vector velocidad con formula tradicional PSO;
            Calcular nueva posición;
        Evaluar fitness ;
Calcular gBest y pBest ;
Ejecutar proceso auto-adaptación de parámetros (Torneo Binario entre 2 partículas al azar);
Reportar resultados;
```

6.6 Experimentos

El rendimiento de oscPSO fue medido optimizando las siguientes funciones mencionadas en [208].

Fueron seleccionadas las siguientes funciones:

- Funciones Unimodales
    - F1: Shifted Sphere Function
    - F2: Shifted Schwefel's Problem 1.2
    - F3: Shifted Rotated High Conditioned Elliptic Function
    - F4: Shifted Schwefel's Problem 1.2 with Noise in Fitness
    - F5: Schwefel's Problem 2.6 with Global Optimum on Bounds
- Multi-Modal Functions
    - F6: Shifted Rosenbrock's Function
    - F7: Shifted Rotated Griewank's Function without Bounds

- F8: Shifted Rotated Ackley's Function with Global Optimum on Bounds
- F9: Shifted Rastrigin's Function
- F10: Shifted Rotated Rastrigin's Function

Todas las funciones fueron optimizadas en 30 dimensiones, considerando un máximo de 300.000 evaluaciones de funciones, de acuerdo a los definido en [208]. En todos los casos, los parámetros $a1$ and $a2$ fueron definidos con un valor de 2 y el factor de inercia w fue reducido linealmente desde 0.9 a 0.4 con el correr de las iteraciones [195] [190].

El rendimiento de ambos algoritmos fue comparado utilizando un análisis de Varianza. Se tomaron las corridas de los mismos y se aplicó el test de Kolmogorov-Smirnov para analizar si las muestras tenían una distribución normal. En aquellos casos donde las muestras mostraron una distribución normal, se calculo el valor estadístico t de Student. Para aquellas muestras que no mostraron una distribución normal, se llevó a cabo la prueba no paramétrica de Kruskal-Wallis. En todos los casos, si el valor de p (p-value) era menos que 0.05, las diferencias fueron consideradas como estadísticamente significantes [118] [115] [119].

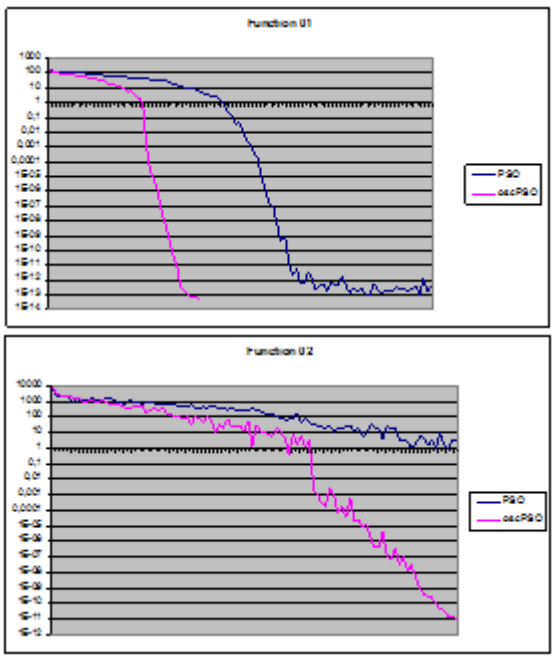
La tabla 6.1 resume el rendimiento del algoritmo comparándolo con la versión canónica del mismo.

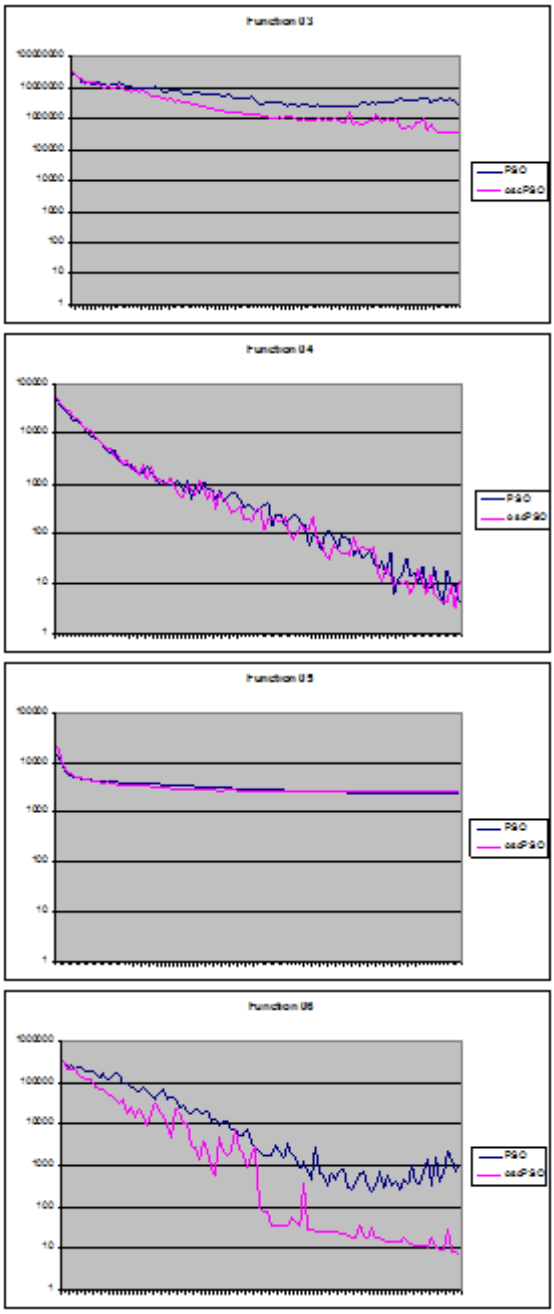
Funcion	PSO		oscPSO	
	Average	Deviation	Average	Deviation
01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
02	3.9327E-06	4.4077E-06	**6.0800E-12**	**2.9361E-11**
03	7.4384E+05	2.3456E+05	**3.3724E+05**	**1.3475E+05**
04	6.3575E-01	5.3254E-01	8.4022E-01	8.7679E-01
05	2.4861E+03	4.6604E+02	2.6997E+03	4.2652E+02
06	7.0806E+01	6.5885E+01	**7.4804E+00**	**1.6092E+01**
07	1.3296E-02	1.0803E+01	1.2017E-02	1.1374E-02
08	2.0953E+01	4.3397E-02	2.0932E+01	6.3025E+00
09	2.4675E+01	5.6866E+00	**2.0973E+01**	**6.2460E+00**
10	1.2545E+02	6.2869E+01	**8.7198E+01**	**4.7837E+01**

Tabla 6.1: Valores promedios de 25 corridas independientes

Como se puede observar en la tabla 6.1, el rendimiento del algoritmo oscPSO fue mejor que la versión original del algoritmo PSO. En 5 de las 10 funciones de pruebas (funciones 2,3,6,9 y 10) mostró mejores resultados. En las cinco funciones restantes, no fue posible determinar cuál de las propuestas tuvo un desempeño superior.

La figuras 6.3 incluye los gráficos que muestran la evolución de la convergencia del algoritmo por cada una de las funciones de prueba. Puede observarse que la ocsPSO converge generalmente más rápido que la versión canónica del algoritmo. Estos mismos gráficos indican que la mayor presión de convergencia debido al uso de procedimiento de búsqueda local PBLD se aplica en el momento correcto del proceso de búsqueda, dado que la convergencia general del algoritmo a óptimos globales se incrementa sin aumentar la tendencia del algoritmo a quedar atrapados en óptimos locales.




Function 03
PSO
cscPSO
Function 04
PSO
cscPSO
Function 05
PSO
cscPSO
Function 06
PSO
cscPSO

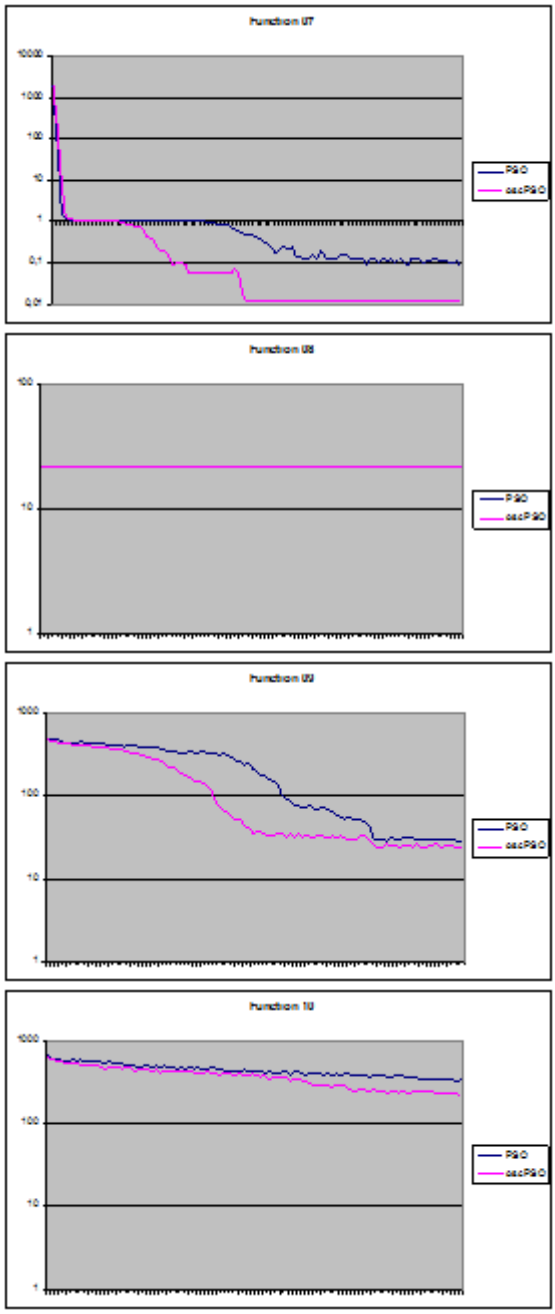


Figura 6.3: Evolución de la convergencia del algoritmo por cada una de las funciones de prueba

6.7 Conclusiones

Este capítulo presentó una nueva variante del algoritmo PSO llamada ocsPSO, que permite mejorar el proceso de búsqueda, particularmente en la fase de aproximación final de las partículas a los valores óptimos. De acuerdo a trabajos previos llevados a cabo por otros autores [210] [211], se demostró que este proceso final de aproximación es ineficiente en la propuesta original del algoritmo PSO, dado que la versión canónica de esta metaheurística utiliza demasiadas iteraciones para lograr la aproximación deseada.

Teniendo esto en mente, surgen dos cuestiones para resolver:

1. Determinar cuándo el algoritmo está oscilando alrededor de algún óptimo local (global).

2. Desarrollar un procedimiento que incremente la eficiencia de la aproximación final de la partícula al óptimo.

Para resolver la primera de las cuestiones, un Detector de Oscilación fue incorporado para analizar la variación del signo del componente velocidad de cada partícula en cada iteración. Cuando se observa que el signo de la velocidad cambia en forma alternativa, se considera que la correspondiente partícula está oscilando para esa dimensión.

Para enfrentar la segunda de las cuestiones detalladas, se incorporó un Procedimiento de Búsqueda Local Determinísta. El propósito de este procedimiento es encontrar la ubicación del óptimo en la menor cantidad de iteraciones posible.

La incorporación de ambos procedimientos requiere la configuración de nuevos parámetros que no están presentes en la versión original del algoritmo. Con el objetivo de no incrementar el nivel de complejidad, se seleccionó un conjunto estándar de valores para esos parámetros, las partículas se inicializaron con diferentes valores para esos parámetros, y se incorporó un proceso de auto adaptación en función de las características de la función a optimizar.

Los resultados reportados muestran que los objetivos propuestos fueron alcanzados, y que el rendimiento de esta nueva versión del algoritmo PSO es mejor que la versión original del mismo.

PSO Binario con Control de Velocidad

7.1 Introducción

Tanto el método PSO binario original como la variante descripta en la sección 4.2 dejan en evidencia la importancia que tiene una adecuada modificación del vector velocidad. En el caso del PSO binario original, el problema central se encuentra en el escaso control que se realiza a la hora de acotarlo. Si el vector velocidad toma valores excesivos, el aplicarle la función sigmoide hace que la probabilidad de cambio sea casi nula. Como forma de compensar este efecto, el método definido en [204] buscó descomponer el vector velocidad en dos partes para poder tener una opción de cambio por el valor contrario al que actualmente tiene la partícula.

Ambos métodos trabajan sobre partículas cuyas posiciones actuales se expresan de manera binaria.

7.2 Algoritmo Propuesto

Esta tesis propone modificar el vector velocidad utilizando una versión modificada del algoritmo gBest PSO continuo. Es decir que, bajo esta propuesta cada partícula tendrá dos vectores velocidad, V1 y V2. El primero se actualiza según la ecuación (7.1).

$$V1_i(t+1)=w.V1_i(t)+\tilde{O}_1.rand_1.(2\times pBesti-1)+\tilde{O}_2.rand_2.(2\times gBest-1) \quad (7.1)$$

donde las variables $rand_1$, $rand_2$, $\tilde{O}_1$ y $\tilde{O}_2$ funcionan de la misma forma que en (?). Los valores p_i y g_i corresponden al i-ésimo dígito binario de los vectores $pBest_i$ y *gBest*, respectivamente.

La diferencia más importante entre las ecuaciones (4.5) y (7.1) es que en la segunda, el desplazamiento del vector *V*1 en las direcciones correspondientes a la mejor solución encontrada por la partícula y al mejor global, no dependen de la posición actual de dicha partícula.

Luego, cada elemento del vector velocidad *v*1 es controlado según (7.2)

$$v1_{ij}(t)=\begin{cases}\delta 1_j & si\ v1_{ij}(t)>\delta 1_j\\ -\delta 1_j & si\ v1_{ij}(t)\leq -\delta 1_j\\ v1_{ij}(t) & en\ otro\ caso\end{cases} \tag{7.2}$$

donde

$$\delta 1_j=\frac{limit1_{sup_j}-limit1_{inf_j}}{2} \tag{7.3}$$

Es decir que, el vector velocidad *v*1 se calcula según la ecuación (7.1) y se controla según la ecuación (7.2). Su valor se utiliza para actualizar el valor del vector velocidad *V*2, como se indica en la ecuación (7.4).

$$v2(t+1)=v2(t)+v1(t+1) \tag{7.4}$$

El vector *V*2 también se controla de manera similar al vector *V*1 cambiando $limit1_{sup_j}$ y $limit1_{inf_j}$ por $limit2_{sup_j}$ y $limit2_{inf_j}$ respectivamente. Esto dará lugar a $´2_j$ que será utilizado como en la ecuación (7.2) para acotar los valores de *V*2. Luego se le aplica la función sigmoide y se calcula la nueva posición de la partícula según la ecuación (4.7).

El concepto de control de velocidad se basa en el método descrito en el capítulo anterior donde se utiliza un control similar para evitar las oscilaciones finales de las partículas alrededor del óptimo.

7.3 Comparación de Rendimiento

En esta sección se compara el rendimiento de la variante de PSO binario propuesta en este trabajo con el método propuesto por Kennedy y Eberhart en [85] y con el PSO binario definido en [204], en la minimización de un conocido conjunto de funciones de prueba *n*-dimensionales las cuales se detallan en la siguiente sección.

Funciones utilizadas

En esta sección se describen las funciones de prueba utilizadas para realizar la evaluación comparativa entre los tres algoritmos.

Función Sphere o Función 1 de De Jong

La definición de la función es la siguiente:

$$f_1(X) = \sum_{i=1}^{n} x_i^2 \tag{7.5}$$

Esta es la función de prueba más sencilla. Es continua, convexa y unimodal. El mínimo global es 0 y se encuentra ubicado en x_i=0 para i=1:n

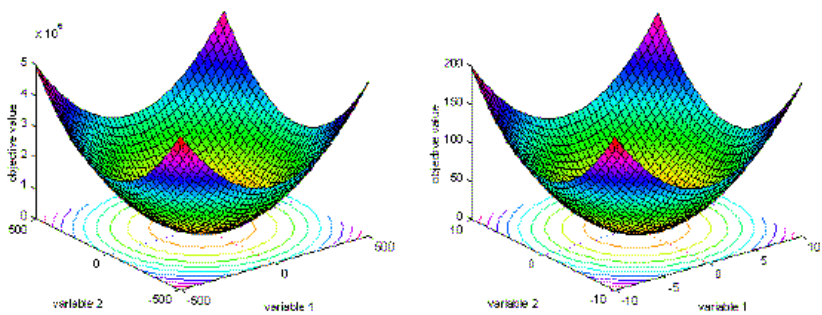


Figura 7.1: Función Sphere utilizando dos dominios distintos. Ambos gráficos se ven muy parecidos pese al cambio de escala. A la izquierda se representa la función haciendo que cada variable tome valores entre -500 y 500 y a derecha se considera un área menor, entre -10 y 10.

Función Rosenbrock o Función 2 de De Jong

En esta función el óptimo global está dentro de un largo y estrecho valle plano con forma parabólica. Si bien encontrar el valle es trivial, la convergencia hacia el óptimo global es difícil y por lo tanto, este problema se ha utilizado en varias ocasiones en evaluar el rendimiento de los algoritmos de optimización.

La definición de la función es:

$$f_2(X)= \sum_{i=1}^{n-1} 100(x_{i+1}-x_i^2)^2+(1-x_i^2) \tag{7.6}$$

El mínimo global es 0 y se encuentra ubicado en x_i=1 para i=1:n

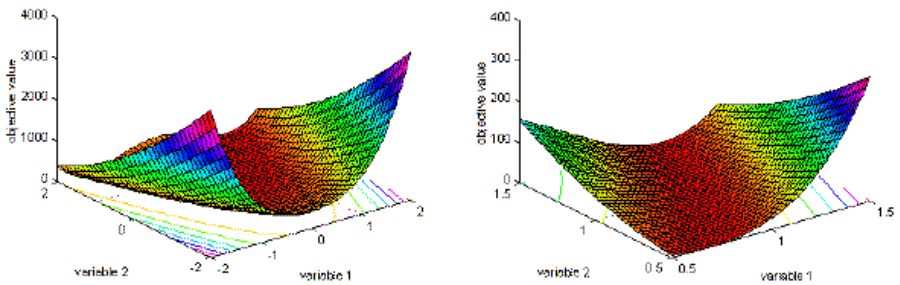


Figura 7.2: Función Rosenbrock. A izquierda se visualiza un área más grande que a la derecha donde se focaliza en el mínimo global

Función Griewangk

Se trata de una función con muchos óptimos locales y se define así:

$$f_3(X)= \sum_{i}^{n} \frac{x(i)^2}{4000}-\Pi_{i=1}^{n}(cos(\frac{x(i)}{sqrt(i)})+1 \tag{7.7}$$

El mínimo global es 0 y se encuentra ubicado en x_i=1 para i=1:n

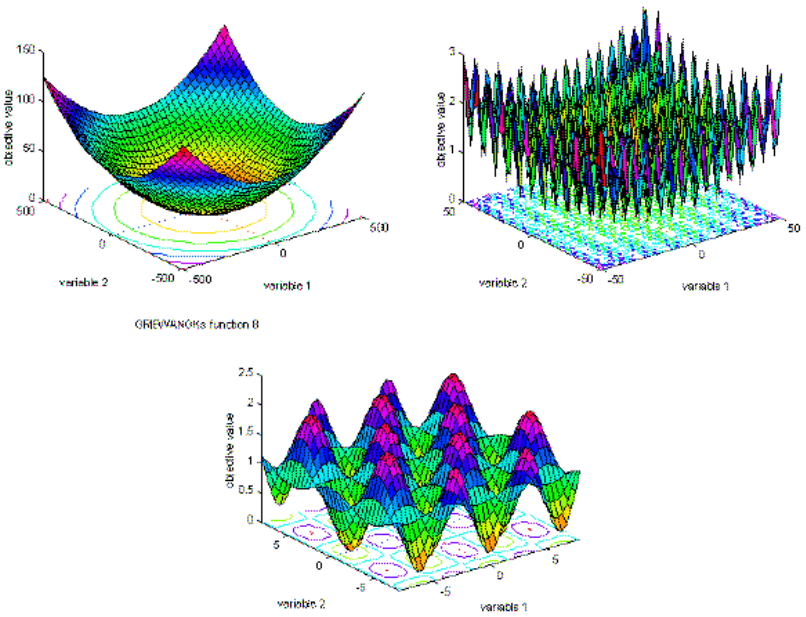


Figura 7.3: Función Griewangk utilizando tres dominios distintos a fin de mostrar las características que posee la función. Arriba a la izquierda se utiliza una área amplia para mostrar que la función es similar a la Función 1. Arriba a la derecha muestra que visto más de cerca posee muchos valles y picos. Sin embargo, la figura inferior muestra que cerca del óptimo los valles y los picos son suaves.

Función Rastrigin

La función se define de la siguiente forma:

$$f_4(X)=10n+\sum_{i=1}^{n}(x_i^2-10cos(2\pi x_i)) \qquad (7.8)$$

Esta función se basa en la función 1 y agrega la función coseno para producir muchos mínimos locales. Esto da lugar a una función multimodal. El mínimo global es 0 y se encuentra ubicado en x_i=0 para i=1:n

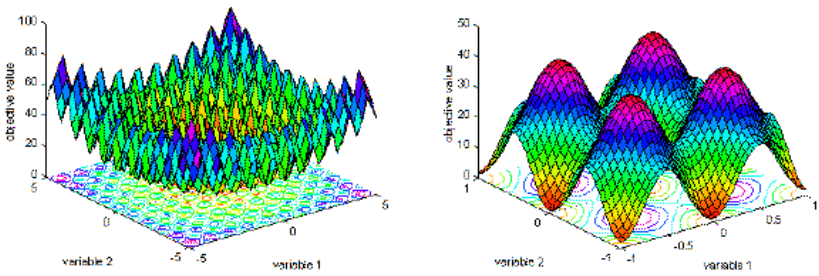


Figura 7.4: Función Rastrigin. A la izquierda sus variables toman valores entre -5 y 5 mientras que a la derecha lo hacen entre -1 y 1

Pruebas realizadas y parámetros utilizados

Se realizaron 40 corridas independientes de cada uno de los métodos utilizando 2000 iteraciones máximas. Se trabajó con n=3, 5 y 10 variables. El tamaño de la población en todos los casos fue de 20 partículas. Los valores de *limite*1 y *limite*2 son iguales para todas las variables; éstos son [0;1] y [0;6] respectivamente. Por lo tanto, los valores de los vectores velocidad *V*1 y *V*2 fueron limitados a los rangos [−0.5,0.5] y [−3,3] respectivamente. Es decir que pueden obtenerse probabilidades en el intervalo [0.0474,0.9526]. Los valores para $\tilde{O}_1$ y $\tilde{O}_2$ fueron establecidos en 0.25. Con respecto a los métodos [85] y [204], se establecieron límites de velocidad entre [−3,3] a fin de mantener el mismo rango de probabilidades.

Resultados obtenidos

La tabla 7.1 muestra el fitness de la mejor solución encontrada por cada método, así como el valor del mejor fitness promedio de las 40 ejecuciones. Las funciones de prueba utilizadas son las siguientes: Sphere, Rosenbrock, Griewangk y Rastrigin, las cuales aparecen numeradas del 1 al 4 respectivamente.

Tabla 7.1: Resultados Obtenidos

		Método Propuesto		PSO binario [85]		PSO binario [204]	
Nro. Var	Nro. Func.	Mejor Fitness	Mejor Fitness Prom.	Mejor Fitness	Mejor Fitness Prom.	Mejor Fitness	Mejor Fitness Prom.
3	1	0	**0**	0	1,2e-09	1,8e-08	6,3e-07
3	2	7,0e-04	**2,8**	1,1e-05	3,0	2,0e-03	5,6
3	3	2,1e-09	**3,3e-03**	2,1e-09	6,8e-03	4,2e-06	8,8e-03
3	4	5,4e-08	**5,4e-08**	5,4e-08	**5,4e-08**	8,9e-06	7,6e-04
5	1	0,0	**3,1e-09**	1,4e-07	1,3e-05	1,7e-04	1,2e-03
5	2	2,2	**28,7**	2,1	111,5	7,2	278,3
5	3	2,6e-09	**8,2e-03**	1,6e-03	2,0e-02	1,9e-02	6,6e-02
5	4	9,0e-08	**1,3e-07**	2,3e-04	5,1e-01	5,0e-01	3,7
10	1	8,2e-05	**9,8e-04**	1,3e-02	7,1e-02	9,7e-02	6,2e-01
10	2	7,3	**141,0**	334,0	2812,8	92013,0	613510,0
10	3	1,3e-02	**7,6e-02**	7,7e-02	1,9e-01	5,2e-01	7,3e-01
10	4	0,8	**4,3**	7,5	15,3	13,7	44,0
20	1	3,3e-01	**8,1e-01**	1,7e+00	3,9e+00	1,2e+01	1,9e+01
20	2	1865,9	**10105**	2,8e+05	1,2e+07	1,2e+08	5,0e+08
20	3	1,4e-01	**2,7e-01**	9,3e-01	1,0e+00	1,3e+00	1,4e+00
20	4	27,7	**43,0**	52,7	88,9	169,8	215,2

Puede observarse que el método propuesto encuentra las mejores soluciones y posee los menores valores de fitness promedio.

La tabla 7.2 indica si los resultados obtenidos son significativos o no. En dicha tabla se ha utilizado el símbolo ▲ para representar que el $p-valor<0.05$, indicando que la hipótesis nula debe ser rechazada. El tipo de test realizado es un test de extremo inferior cuya hipótesis nula afirma que la media del método indicado en la fila de la tabla no es menor a la media del método indicado en la columna. El símbolo ▽ indica que la hipótesis nula no se rechaza.

La figura 7.5 muestra los diagramas de caja calculados sobre los mejores resultados obtenidos en cada una de las 40 corridas. Cada columna corresponde a una función distinta. Los diagramas de una misma fila corresponden a la misma cantidad de variables. Considerando de arriba hacia abajo, las filas 1, 2, 3 y 4 corresponden a los resultados obtenidos al evaluar las funciones en 3, 5, 10 y 20 variables respectivamente. En cada figura, los métodos están

numerados del 1 al 3 correspondiendo a el método propuesto, PSO binario [85] y PSO binario [204] respectivamente.

Como puede observarse, efectivamente, el método propuesto ofrece mejores soluciones que las otras dos alternativas de PSO.

La figura 7.6 está organizada de la misma forma que la figura 7.5 y muestra los diagramas de caja correspondientes al fitness promedio de cada una de las 40 corridas realizadas para cada función y para cada cantidad de variables consideradas. En ella puede observarse la diversidad poblacional de cada método. En general, a partir de la altura de cada caja, puede decirse que si bien el método [204] presenta los mayores rangos intercuartiles, las soluciones que ofrece son las peores. En cuanto al método propuesto y el PSO binario [85], los rangos son equivalentes.

Tabla 7.2: Resultados de las pruebas de hipótesis. El símbolo ▲ representa que el p−valor fue inferior a 0.05 es decir que se rechaza la hipótesis nula que afirma que la media del método indicado en la fila no es menor a la del método indicado en la columna

nro Var.	Método	Propuesto	PSO binario	PSO binario [85]				PSO binario [204]			
3	PSO propuesto			▽	▽	▽	▽	▲	▽	▲	▽
3	PSO binario [85]	▽	▽	▽	▽			▲	▽	▽	▽
3	PSO binario [204]	▽	▽	▽	▽	▽	▽	▽	▽		
5	PSO propuesto			▽	▽	▲	▽	▲	▲	▲	▲
5	PSO binario [85]	▽	▽	▽	▽			▲	▽	▲	▲
5	PSO binario [204]	▽	▽	▽	▽	▽	▽	▽	▽		
10	PSO propuesto			▲	▲	▲	▲	▲	▲	▲	▲
10	PSO binario [85]	▽	▽	▽	▽			▲	▲	▲	▲
10	PSO binario [204]	▽	▽	▽	▽	▽	▽	▽	▽		
20	PSO propuesto			▲	▲	▲	▲	▲	▲	▲	▲
20	PSO binario [85]	▽	▽	▽	▽			▲	▲	▲	▲
20	PSO binario [204]	▽	▽	▽	▽	▽	▽	▽	▽		

7.4 Conclusiones

En este capítulo se ha presentado una variante del método PSO binario propuesto originalmente en [85] que controla las modificaciones del vector velocidad utilizando una variante del método PSO continuo.

Los resultados obtenidos al minimizar un conjunto de funciones de prueba son mejores a los arrojados por los métodos definidos en [85] y [204].

La tabla 7.2 permite ver que a medida que aumenta la cantidad de variables utilizadas, la diferencia de medias se hace cada vez más significativa.

La figura 7.5 permite mostrar que en la mayoría de las ejecuciones, los resultados obtenidos por el método propuesto han sido superiores a los de los otros dos. Asimismo, en la figura 7.6 se observa que el método propuesto es el que genera los mejores resultados de fitness promedio de la población, por lo menos en las funciones de prueba evaluadas. Si se considera el rango intercuartil del fitness promedio (amplitud de los diagramas de caja de la figura 2) puede afirmarse que el método propuesto en este articulo y el PSO binario de [85] son equivalentes, ofreciendo el primero las mejores soluciones.

Es de sumo interés avanzar con el análisis del algoritmo propuesto. Por un lado, midiendo el rendimiento cuando el número de dimensiones del problema crece, con el objetivo de aplicarlo a la resolución de problemas del mundo real. Por otro lado, analizando su desempeño utilizando PSO de población variable [212]. Esto último permitiría adaptar la cantidad de individuos del cúmulo (*swarm*) en función de la complejidad del problema a resolver.

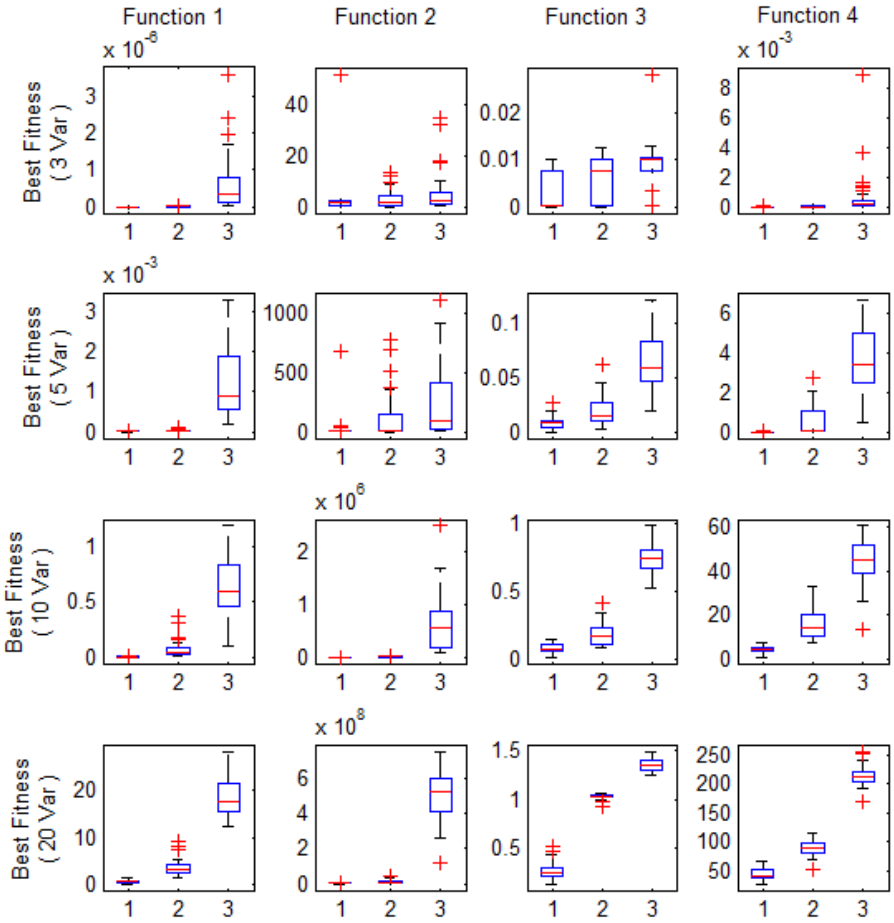


Figura 7.5: Diagramas de caja correspondientes a las mejores soluciones obtenidas en cada una de las 40 corridas independientes. Sobre el eje de las abscisas se indica el método: 1 = Método propuesto, 2 = PSO binario [85] y 3 = PSO binario [204]. Cada fila indica los resultados obtenidos con 3, 5, 10 y 20 variables

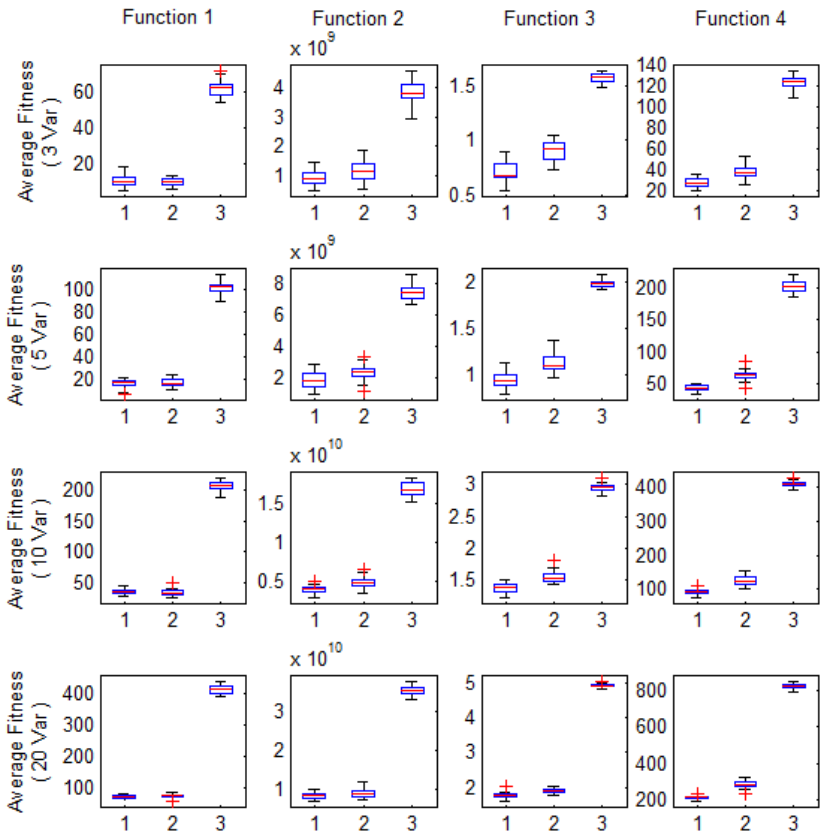


Figura 7.6: Diagramas de caja correspondientes al fitness promedio de cada una de las 40 corridas independientes. Sobre el eje de las abscisas se indica el método: 1 = Método propuesto, 2 = PSO binario [85] y 3 = PSO binario [204]. Cada fila indica los resultados obtenidos con 3, 5, 10 y 20 variables.

CAPÍTULO 8

varMOPSO - MOPSO de Población Variable

8.1 Introducción

Se han desarrollado múltiples técnicas, generalmente surgidas del ámbito de la ingeniería, para tratar con problemas con más de un objetivo [213], pero podemos citar el trabajo de Rosenberg [214] de mediados de los 60's como pionero en la utilización de algoritmos evolutivos para la resolución de este tipo de problemas.

Si bien esta área de interés se mantuvo relativamente inexplorada por los siguientes 25 años, en la actualidad este cuerpo de conocimiento atrae el trabajo de muchos científicos dedicados al tópico de optimización y aprendizaje de máquinas [137].

La primera implementación de un algoritmo evolutivo aplicado a optimización multi-objetivo fue el algoritmo VEGA (*Vector Evaluation Genetic Algorithms*), producto de la tesis doctoral de J. David Schaffer y presentado en la conferencia inicial sobre algoritmos genéticos en 1985 [215] [216] [217].

Un problema multi-objetivo implica la optimización simultánea de varias funciones objetivo, normalmente en conflicto. En este tipo de problemas, la solución consiste en un conjunto de soluciones óptimas. La razón de crear un conjunto de soluciones se debe a que no se puede considerar ninguna de ellas como la mejor de todas cuando se consideran todas las funciones objetivo, dado que la mejor solución para una de las funciones objetivo puede ser peor para otra de las funciones. En este contexto, se utiliza el concepto de dominancia de Pareto. Dada una población *S*, el conjunto de soluciones óptimas está definido como el sub-conjunto de *S* formado por todos los individuos de *S* que son no dominados. En el espacio de soluciones, este conjunto es conocido como el conjunto de Pareto, y su imagen en el espacio de las funciones objetivo es conocido como el Frente de Pareto (ver capítulo 3, Optimización Multi-objetivo, para más detalles).

El propósito principal de un algoritmo de optimización multi-objetivo es encontrar las soluciones candidatas tan cerca como sea posible del

frente de Pareto real del problema, que estén uniformemente distribuidas y que además, maximicen la cobertura del mismo [218].

La metaheuristica PSO ha sido utilizada para resolver problemas multi-objetivo reportándose buenos resultados [219] [156] [220] [221]. En un algoritmo del tipo MOPSO (*Multi-objective Particle Swarm Optimization*) la selección del líder tiene gran influencia tanto en lo relacionado con la convergencia de las soluciones al frente de Pareto así como en la diversidad de las soluciones encontradas [222]. De acuerdo a [223], es necesario que el algoritmo posea un mecanismo de elitismo para converger al frente óptimo de Pareto. En la mayoría de las implementaciones de algoritmos de la clase MOPSO, el conjunto de soluciones no dominadas es un archivo externo con una cantidad fija de soluciones. Existen diferentes métodos para seleccionar qué soluciones no dominadas eliminar cuando la cantidad de miembros del conjunto exceden la cantidad pre-fijada [224] [223] [220] [222]. Cuando la ejecución del algoritmo finaliza, este archivo externo contiene el conjunto de soluciones del frente de Pareto obtenido. Idealmente, este archivo es un subconjunto de las soluciones del frente de Pareto real para el problema optimizado.
En este capítulo de la presente tesis se propone una nueva versión de un algoritmo PSO aplicado a problemas multi-objetivo, se efectúa una descripción detallada del mismo, y se realiza una comparación entre el rendimiento de la versión propuesta y los algoritmos más destacados en esta clase de optimización.

8.2 Algoritmo Propuesto (varMOPSO)

Existen mas de 30 variaciones que permiten aplicar la metaheuristica PSO, propuesta por Kennedy y Eberhart [84], a problemas multi-objetivo. Entre estas variaciones de algoritmos tipo MOPSO, se encuentran como los más destacados SMPSO y OMOPSO dado que los mismos presentan los mejores indicadores de rendimiento [220].

En este capítulo se presenta una nueva propuesta de algoritmo tipo MOPSO, llamado varMOPSO, que incorpora el concepto de población variable. De la misma forma que la solución adoptada en [222], este algoritmo utiliza un archivo externo para mantener las soluciones no dominadas encontradas. Para calcular la velocidad de las partículas se utiliza la ecuación de velocidad con constricción propuesta en [156], basada en el factor de constricción desarrollado en [193].

La variación de la población entre generaciones es el resultado de agregar el concepto de edad para las partículas y la utilización de un procedimiento de inserción / eliminación basado en la habilidad de cada individuo para resolver el problema presentado. El algoritmo utiliza el concepto de elitismo, y sólo las soluciones dominadas se eliminan de la población principal. Estos procedimientos están basados en la versión de PSO mono-objetivo presentada en [212].
Para los algoritmos PSO, la selección del líder es crucial cuando se calcula la velocidad de las partículas [222] [180] [211]. El método propuesto aquí utiliza diferentes mecanismos de selección de líderes entre las dos fases del algoritmo. Durante la primera fase, se seleccionan dos líderes al azar del archivo externo de soluciones no dominadas, y se realiza un torneo binario basado en su valor de agrupamiento (*crowding value*), de acuerdo al mecanismo propuesto en [225]. Durante la fase final del proceso, el procedimiento de selección detallado anteriormente se utiliza el 50% de las veces, mientras que el restante 50% se elige en forma determinística el individuo con menor valor de agrupamiento, con el objetivo de encontrar nuevas soluciones en las regiones menos pobladas del frente de Pareto. Para evaluar el rendimiento del método propuesto se realizaron comparaciones con tres metaheuristicas representativas del estado del arte en optimización multi-objetivo, NSGAII [154], SPEA2 [155] y SMPSO [156].
El experimento y la obtención de resultados se llevó a cabo utilizando el framework de desarrollo jMetal [226], utilizándose la implementación de las metaheurísticas anteriormente detalladas, e incorporando en el mismo el método propuesto.
En la siguiente sección se describe en detalle el algoritmo propuesto. Luego, en las subsiguientes dos secciones se explican los experimentos llevados a cabo, los resultados y las conclusiones obtenidas.

Descripción del Algoritmo

La principal contribución de esta propuesta está focalizada en la variación del tamaño de la población principal del algoritmo. Esta incorporación permite al algoritmo adaptar el número de individuos que utiliza en ambas fases del proceso de búsqueda, exploración y explotación, basada en la dificultad del problema.
La fase de exploración del algoritmo es la fase inicial donde se efectúa la búsqueda de las regiones promisorias del espacio de búsqueda. En esta fase, la población crece hasta 6 o 7 veces el tamaño de la población inicial, dependiendo del problema y la ejecución. Entonces,

a medida que la iteraciones del algoritmo avanzan, el número de individuos es gradualmente reducido hasta alcanzar un tamaño mínimo de población pre-establecido. Esta última etapa de ejecución, cuando la población alcanza un valor mínimo fijado a priori, se identifica como la fase de explotación del algoritmo.
En los siguientes párrafos, los conceptos y procedimientos incorporados al algoritmo para cambiar el tamaño de la población son descritos en detalle.

Tiempo de Vida

El tiempo de vida de las partículas en uno de los conceptos fundamentales de esta versión propuesta. El mismo define la permanencia de la partícula dentro de la población. Este valor está expresado en cantidad de iteraciones, que una vez alcanzadas, habilita la remoción de la partícula. Esta última se elimina de la población, a menos que la misma pertenezca al conjunto de soluciones no dominadas. El tiempo de vida de una partícula está íntimamente relacionado con su valor de aptitud y permite que las mejores permanezcan más tiempo en la población, influenciando así el comportamiento del resto de las partículas.
Para evaluar y asignar el tiempo de vida de cada individuo de la población, los miembros de la misma se agrupan de acuerdo a un valor de ranking en *k* clases. El valor de ranking indica el nivel de no dominancia de los individuos dentro de la población, calculado de acuerdo a la propuesta del algoritmo NSGAII, en [154]. Con el resultado de este agrupamiento, se aplica el método propuesto en [212].
Para este procedimiento se aplica elitismo. La solución es removida de la población si y sólo si no es una solución no dominada. Este procedimiento se ejecuta mientras el tamaño de la población sea mayor que un valor mínimo pre-establecido.

Inserción de Partículas

El procedimiento de inserción de partículas compensa el procedimiento de remoción de las mismas, incrementa la capacidad de búsqueda y la velocidad de convergencia. La cantidad de partículas a incorporar en cada iteración se corresponde con la cantidad de individuos aislados. Un individuo aislado es una partícula que no tiene ningún vecino dentro de un radio *r* calculado de acuerdo al criterio definido en [212], y presentado en la siguiente ecuación:

$$d_i = min\{\|x_i - x_j\|; \forall j \quad x_i, x_j \in S; x_i \neq x_j\} \; i = 1..n \quad ; \quad r = \sum_{i=1}^{n} d_i / n \tag{8.1}$$

El mecanismo adoptado para definir la ubicación de los nuevos individuos es el siguiente: 20% de estas nuevas partículas reciben el vector posición de alguno de lo mejores individuos de la población, y su vector velocidad se define en forma aleatoria; el restante 80% tanto su ubicación como su velocidad son aleatorios, de acuerdo a lo definido en [212].

Para seleccionar los mejores individuos se eligen dos partículas al azar del archivo externo de soluciones no dominadas, y se efectúa un torneo binario entre ellas. Se define como ganadora la solución que se encuentra en la zona menos poblada dentro del espacio de funciones objetivo [154].

Para prevenir que el tamaño de la población crezca en forma excesiva, existe una variable *c* que regula el crecimiento de la misma. Esta variable refleja la probabilidad de crear una nueva partícula. Al comienzo de la ejecución del algoritmo, esta variable tiene un valor de 1.0. Si el tamaño de la población crece más que un 5% entre generaciones, el valor de la variable *c* decrece 0,1. Cuando esta variable *c* alcanza el valor de 0, permanece con ese valor por el resto de la iteraciones de la ejecución del algoritmo. Es una variable cuyos valores son monótonos decrecientes acorde al número de iteración. De esta forma, el algoritmo incrementa el tamaño de la población durante las primeras iteraciones, alcanzado su valor máximo en esta primera fase de ejecución. Una vez que la variable *c* alcanza el valor 0, el tamaño de la población decrece en forma gradual hasta alcanzar la cantidad mínima de individuos pre-establecida.

Remoción de partículas

Este procedimiento tiene como objetivo eliminar las partículas que se encuentran en áreas superpobladas del espacio de parámetros (espacio de búsqueda).

En una primera instancia, se calcula la cantidad de vecinos cercanos de cada partícula. Este número se corresponde con la cantidad de partículas que rodean a un individuo dentro de un radio r_1, calculado de acuerdo a la siguiente ecuación:

$$r_1 = \sum_{i=1}^{n} (r - \|x_i - x_j\|)/n \quad ; \quad \forall j \quad x_i, x_j \in S \; ; \; x_i \neq x_j \quad i = 1..n \tag{8.2}$$

Todas las partículas con más de 0 vecinos cercanos son candidatas para remoción. Para cada individuo, se calcula un valor normalizado entre 0 y 1 considerando dos factores, la aptitud de la partícula para resolver el problema y la densidad de los individuos que rodean cada solución. La aptitud de cada partícula se calcula de acuerdo a la siguiente ecuación:

$$a_i = (max(ra) - ra_i)/(max(ra) - min(ra)) \; ; \; p_i = (pr_i / max(pr))$$

$$ap_i = 0.5 \times a_i + 0.5 \times p_i \; ; \; i = 1..n \tag{8.3}$$

donde

- $max(ra)$ = nivel de dominancia máximo dentro de el sobconjunto de partículas que son candidatas para ser removidas;
- ra_i = nivel de dominancia de cada partícula;
- $max(pr)$ = máxima cantidad de vecinos cercanos en la subpoblación de partículas considerada que son candidatas para la remoción
- pr_i = número de vecinos cercanos de cada partícula.

El calculo previo siempre retorna un valor entre 0 y 1. Como se puede inferir analizando la ecuación ?, el cálculo de la aptitud normalizada de cada partícula depende en un 50% del ranking de la misma (nivel de no dominación dentro de la población) y en un 50% de la cantidad de vecinos dentro del radio r_1 previamente descrito, calculado en el espacio de búsqueda.

Por cada partícula, un valor aleatorio uniformemente distribuido entre 0 y 0.50 es seleccionado. Si el valor de aptitud normalizado es menor que este valor, la partícula es removida de la población, en otro caso, el individuo permanece en la misma. La probabilidad de remover una partícula dada es inversamente proporcional al valor calculado de aptitud normalizada.

Este procedimiento utiliza elitismo. La solución es removida de la población sólo si no es una solución no dominada. Este proceso de remoción de partículas es ejecutado mientras el tamaño de la población total sea mayor que un valor mínimo pre-establecido.

El algoritmo 7 muestra el detalle del pseudo-código de la presente propuesta.

El proceso *ComputeLifeTimes* recibe la población completa de partículas y computa el tiempo de vida correspondiente a las partículas que tienen tiempo de vida nulo.

El proceso *AddArchiveLeaders* agrega soluciones no dominadas al archivo externo correspondiente, y elimina soluciones dominadas por estos nuevos integrantes. Si el archivo de soluciones externas tiene una cantidad mayor de soluciones que un valor fijo pre-definido, esta función utiliza el concepto de *crowding distance* del algoritmo NSGAII [227] para decidir cuáles soluciones deben permanecer y cuales deben eliminarse.

Algorithm 7: Algoritmo varMOPSO propuesto

```
Pop = CreatePopulation(N) {se genera la población inicial} ;
ComputeFitness(Pop);
ComputeRank(Pop) {se computa el nivel de no dominancia de cada partícula};
ComputeLifeTimes(Pop);
AddArchiveLeaders (Ext);
ComputeRadius(Pop, cant, Eliminados) {la primera vez, Eliminados está vacío};
while no condición de fin no es alcanzada do
    NewPop = CreateNewParticles(Pop, cant);
    ComputeLifeTimes(NewPop);
    Pop = Pop ∪ NewPop;
    ComputeSpeed(Pop);
    ComputeNewPosition(Pop);
    ComputeFitness(Pop);
    AddArchiveLeaders(Ext);
    ComputeRank(Pop);
    Se resta 1 al tiempo de vida de cada partícula;
    Se remueven las partículas con tiempo de vida nulo;
    ComputeRadius(Pop, cant, Eliminados);
    EliminCrow(Eliminados);
    ComputeLifeTimes(Pop);
Output : Ext;
```

El cómputo de los valores de radio, de acuerdo a ecuación (8.1) y (8.2) se efectúa dentro de la rutina *ComputeRadius*. La misma recibe el enjambre completo de partículas como parámetro de entrada y

devuelve la cantidad de partículas que deben ser insertadas en la población. Este módulo es el encargado de evitar la concentración de muchas partículas en las mismas regiones del espacio de búsqueda, por esta razón también retorna en *Eliminados* la lista de partículas que tienen vecinos cercanos. Estas partículas son eliminadas en la rutina *EliminCrow*, basado en el valor de aptitud normalizado calculado para cada individuo de acuerdo a la ecuación (8.3).

En la figura 8.1 se puede apreciar la evolución del tamaño de la población para una corrida típica. Al principio, la población aproximadamente sextuplica su tamaño, y luego disminuye hasta alcanzar el límite inferior prefijado. En la figura 8.2 se visualiza la evolución del frente de Pareto obtenido por una corrida del algoritmo. En la imagen de la izquierda se aprecia el frente de Pareto obtenido en el momento en que el algoritmo reduce la cantidad de individuos al mínimo prefijado. En la imagen de la derecha se muestra el frente de Pareto obtenido. Comparando ambas imágenes es posible inferir que durante el proceso inicial el algoritmo identifica las zonas promisorias, y durante el proceso final de la ejecución, al alcanzar la población mínima, se refinan las soluciones encontradas. En esta última etapa el algoritmo actúa como una especie de procedimiento de búsqueda local.

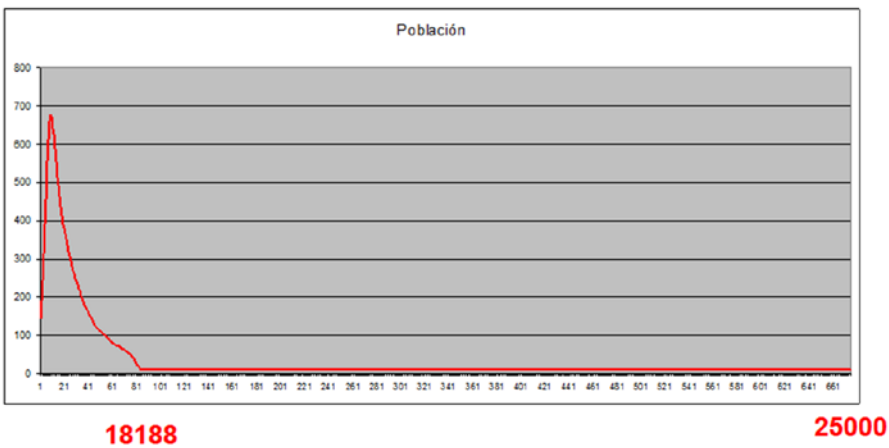


Figura 8.1: Evolución del tamaño de la población en varMOPSO. Función ZDT6

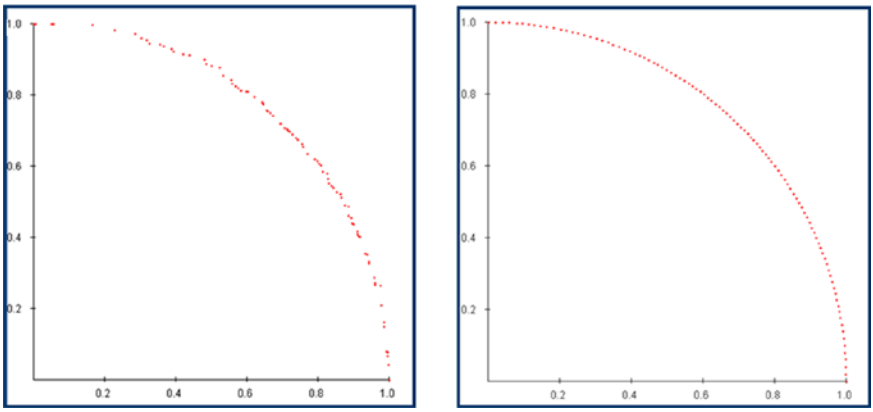


Figura 8.2: Evolución del frente de Pareto. Etapa de exploración - explotación. Función DTLZ3

8.3 Experimentos

Para evaluar el rendimiento de varMOPSO, se utilizaron el conjunto de funciones de prueba ZDT (Zitzler-Deb-Thiele) [228] y DTLZ (Deb-Thiele-Laumanns-Zitzler) [229] configuradas con 2 objetivos. Los experimentos fueron llevados a cabo con los algoritmos NSGAII, SPEA2, SMPSO y varMOPSO.

Los siguientes indicadores fueron calculados: *Additive Unary Epsilon*, *Spread*, e *Hipervolumen* [150][230][231], para comparar la calidad de los frentes de Pareto obtenidos por cada uno de los algoritmos. Estos indicadores son representativos para medir respectivamente la convergencia general al frente de Pareto real, la cobertura del frente de Pareto obtenido y la uniformidad alcanzada en la distribución de las soluciones [230] [231].

Para desarrollar el algoritmo así como para llevar adelante los experimentos y generar los resultados, como se mencionó anteriormente se utilizó el framework de desarrollo jMetal [226], dada su facilidad de uso y su buena documentación.

Para los primeros tres algoritmos se utilizó una población fija de 100 individuos. Para el caso de varMOPSO, se utilizó una población inicial de 100 partículas. El tamaño del archivo externo de soluciones no dominadas fue de 100 en todos los casos, para permitir una comparación justa de los frente de Pareto obtenidos.

En NSGAII [227] y SPEA2 [232] se utilizó SBX y mutación polinomial como operadores de cruza y mutación, respectivamente. La

probabilidad de cruza se configuró con el valor 0,9 y la probabilidad de mutación en 1/L, donde L es el número de variables de decisión. En SMPSO [233], se utilizó mutación polinomial. La probabilidad de mutación se definió en 1/L, donde L es el número de variables de decisión. En varMOPSO, un mínimo de población de 10 individuos y k=4 clases fueron utilizados para calcular el tiempo de vida de las partículas.

En todos los casos se aplicaron 25.000 evaluación de función, y cada algoritmo fue ejecutado 25 veces para cada problema.

La Tabla 8.1 muestra los resultados del indicador *Hipervolumen* para todos los algoritmos aplicados a todos los problemas de prueba. La tabla incluye la mediana y el valor del rango intercuartil (IQR). Los mejores resultados se muestran en gris oscuro, y los segundos mejores resultados en gris claro. Las tablas 8.2 y 8.3 muestran los resultados del test de hipótesis e indican si los valores son estadísticamente significantes para los dos grupos de las funciones de prueba. En cada tabla, el símbolo $\triangledown$ y $\blacktriangle$ implica un p-value < 0.05, indicando que la hipótesis nula (ambas distribuciones tienen la misma mediana) es rechazada; en otro caso se utiliza el símbolo −.

En forma similar, las tablas 8.4, 8.5y 8.6 muestran la misma información para el indicador *Spread*, y las tablas 8.7, 8.8 y 8.9 muestran la Mediana y el rango intercuartil del indicador *Epsilon* (I_{μ}).

Las figuras 8.3 a 8.8 muestran los diagramas de cajas para los diferentes indicadores calculados.

Tabla 8.1: Hipervolumen. Mediana e Intercuatil

	NSGAII	SPEA2	varMOPSO	SMPSO
ZDT1	$6{,}59e-01_{3{,}8e-04}$	$6{,}60e-01_{3{,}7e-04}$	$6{,}62e-01_{3{,}3e-04}$	$6{,}62e-01_{1{,}1e-04}$
ZDT2	$3{,}26e-01_{2{,}7e-04}$	$3{,}26e-01_{6{,}7e-04}$	$3{,}29e-01_{2{,}5e-01}$	$3{,}29e-01_{1{,}6e-04}$
ZDT3	$5{,}15e-01_{2{,}0e-04}$	$5{,}14e-01_{4{,}9e-04}$	$5{,}16e-01_{3{,}0e-04}$	$5{,}15e-01_{6{,}1e-04}$
ZDT4	$6{,}54e-01_{6{,}5e-03}$	$6{,}51e-01_{7{,}4e-03}$	$6{,}62e-01_{4{,}0e-04}$	$6{,}61e-01_{2{,}4e-04}$
ZDT6	$3{,}89e-01_{1{,}9e-03}$	$3{,}79e-01_{4{,}6e-03}$	$4{,}01e-01_{1{,}2e-04}$	$4{,}01e-01_{1{,}2e-04}$
DTLZ1	$4{,}87e-01_{7{,}8e-03}$	$4{,}85e-01_{5{,}7e-03}$	$4{,}95e-01_{2{,}0e-04}$	$4{,}94e-01_{2{,}9e-04}$
DTLZ2	$2{,}11e-01_{3{,}1e-04}$	$2{,}12e-01_{1{,}8e-04}$	$2{,}12e-01_{2{,}6e-04}$	$2{,}12e-01_{1{,}7e-04}$
DTLZ3	$0{,}00e+00_{3{,}8e-02}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}12e-01_{1{,}6e-04}$	$2{,}12e-01_{1{,}3e-01}$
DTLZ4	$2{,}09e-01_{2{,}1e-01}$	$2{,}10e-01_{2{,}1e-01}$	$2{,}10e-01_{2{,}6e-04}$	$2{,}10e-01_{1{,}2e-04}$
DTLZ5	$2{,}11e-01_{3{,}3e-04}$	$2{,}12e-01_{1{,}5e-04}$	$2{,}12e-01_{2{,}8e-04}$	$2{,}12e-01_{1{,}5e-04}$
DTLZ6	$1{,}73e-01_{3{,}9e-02}$	$8{,}12e-03_{1{,}2e-02}$	$2{,}12e-01_{2{,}1e-01}$	$2{,}12e-01_{6{,}9e-05}$
DTLZ7	$3{,}33e-01_{2{,}3e-04}$	$3{,}34e-01_{2{,}7e-04}$	$3{,}34e-01_{5{,}4e-05}$	$3{,}34e-01_{1{,}1e-04}$

Tabla 8.2: Hipervolumen. Significancia estadística en funciones ZDT

	SPEA2					varMOPSO					SMPSO				
NSGAII	▽	–	▲	–	▲	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
SPEA2						▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
varMOPSO											–	–	▲	▲	–

Tabla 8.3: Hipervolumen. Significancia estadística en funciones DTLZ Functions

	SPEA2							varMOPSO							SMPSO						
NSGAII	–	▽	–	▽	▽	▲	▽	▽	▽	▽	▽	▽	–	▽	▽	▽	▽	▽	▽	▽	▽
SPEA2								▽	▽	▽	▽	▽	–	▽	▽	▽	▽	▽	▽	▽	▽
varMOPSO															▲	–	▲	▲	▲	▽	▲

Tabla 8.4: SPREAD. Mediana e Intercuartil

	NSGAII	SPEA2	varMOPSO	SMPSO
ZDT1	$3{,}69e-01_{5{,}8e-02}$	$1{,}50e-01_{2{,}3e-02}$	$8{,}32e-02_{2{,}5e-02}$	$7{,}83e-02_{1{,}2e-02}$
ZDT2	$3{,}71e-01_{3{,}1e-02}$	$1{,}56e-01_{1{,}8e-02}$	$8{,}10e-02_{3{,}8e-01}$	$7{,}53e-02_{1{,}9e-02}$
ZDT3	$7{,}47e-01_{2{,}6e-02}$	$7{,}08e-01_{8{,}6e-03}$	$7{,}09e-01_{1{,}2e-02}$	$7{,}09e-01_{9{,}8e-03}$
ZDT4	$3{,}93e-01_{4{,}4e-02}$	$2{,}82e-01_{1{,}1e-01}$	$8{,}50e-02_{1{,}3e-02}$	$9{,}04e-02_{1{,}5e-02}$
ZDT6	$3{,}64e-01_{4{,}9e-02}$	$2{,}26e-01_{2{,}1e-02}$	$9{,}77e-02_{6{,}8e-01}$	$8{,}80e-02_{8{,}9e-01}$
DTLZ1	$3{,}93e-01_{5{,}7e-02}$	$1{,}84e-01_{7{,}0e-02}$	$6{,}82e-02_{1{,}3e-02}$	$7{,}29e-02_{2{,}3e-02}$
DTLZ2	$3{,}74e-01_{5{,}3e-02}$	$1{,}51e-01_{1{,}9e-02}$	$1{,}30e-01_{1{,}4e-02}$	$1{,}25e-01_{2{,}2e-02}$
DTLZ3	$8{,}95e-01_{1{,}5e-01}$	$1{,}05e+00_{1{,}6e-01}$	$1{,}13e-01_{2{,}5e-02}$	$1{,}41e-01_{7{,}7e-01}$
DTLZ4	$3{,}97e-01_{6{,}2e-01}$	$1{,}54e-01_{8{,}6e-01}$	$1{,}14e-01_{3{,}7e-02}$	$1{,}23e-01_{2{,}7e-02}$
DTLZ5	$3{,}80e-01_{4{,}1e-02}$	$1{,}52e-01_{2{,}0e-02}$	$1{,}30e-01_{2{,}6e-02}$	$1{,}28e-01_{1{,}2e-02}$
DTLZ6	$8{,}67e-01_{3{,}6e-01}$	$8{,}16e-01_{1{,}2e-01}$	$1{,}09e-01_{3{,}6e-01}$	$1{,}01e-01_{2{,}7e-02}$
DTLZ7	$6{,}31e-01_{2{,}3e-02}$	$5{,}47e-01_{1{,}2e-02}$	$5{,}19e-01_{1{,}3e-03}$	$5{,}19e-01_{2{,}4e-03}$

Tabla 8.5: Spread. Significancia estadística en funciones ZDT

	SPEA2					varMOPSO					SMPSO				
NSGAII	▽	▽	▽	▽	▽	▽	▽	▽	▽	–	▽	▽	▽	▽	▽
SPEA2						▽	▽	–	▽	–	▽	▽	–	▽	–
varMOPSO											–	–	–	–	–

Tabla 8.6: Spread. Significancia estadística en funciones DTLZ

	SPEA2							varMOPSO							SMPSO						
NSGAII	▽	▽	▲	▽	▽	–	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
SPEA2								▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
varMOPSO															–	–	▲	–	–	–	–

Tabla 8.7: Epsilon. Mediana e Intercuartil

	NSGAII	SPEA2	varMOPSO	SMPSO
ZDT1	$1{,}26e-02_{2{,}1e-03}$	$8{,}88e-03_{6{,}3e-04}$	$5{,}76e-03_{4{,}7e-04}$	$5{,}60e-03_{1{,}5e-04}$
ZDT2	$1{,}32e-02_{2{,}6e-03}$	$9{,}04e-03_{1{,}2e-03}$	$5{,}63e-03_{5{,}4e-01}$	$5{,}58e-03_{2{,}9e-04}$
ZDT3	$7{,}98e-03_{1{,}8e-03}$	$9{,}90e-03_{2{,}5e-03}$	$5{,}45e-03_{9{,}1e-04}$	$5{,}73e-03_{1{,}2e-03}$
ZDT4	$1{,}62e-02_{3{,}9e-03}$	$3{,}47e-02_{5{,}6e-02}$	$5{,}93e-03_{6{,}0e-04}$	$6{,}13e-03_{5{,}8e-04}$
ZDT6	$1{,}43e-02_{2{,}5e-03}$	$2{,}42e-02_{5{,}0e-03}$	$4{,}76e-03_{4{,}2e-04}$	$4{,}68e-03_{3{,}3e-04}$
DTLZ1	$7{,}69e-03_{2{,}3e-03}$	$6{,}14e-03_{2{,}8e-03}$	$2{,}91e-03_{2{,}1e-04}$	$3{,}03e-03_{1{,}8e-04}$
DTLZ2	$1{,}18e-02_{3{,}3e-03}$	$6{,}92e-03_{1{,}1e-03}$	$5{,}05e-03_{3{,}1e-04}$	$5{,}28e-03_{1{,}7e-04}$
DTLZ3	$9{,}34e-01_{1{,}5e+00}$	$2{,}36e+00_{1{,}2e+00}$	$5{,}34e-03_{3{,}6e-04}$	$5{,}51e-03_{7{,}0e-01}$
DTLZ4	$1{,}08e-02_{9{,}9e-01}$	$8{,}60e-03_{9{,}9e-01}$	$5{,}42e-03_{4{,}3e-04}$	$5{,}39e-03_{3{,}7e-04}$
DTLZ5	$1{,}06e-02_{2{,}1e-03}$	$7{,}68e-03_{1{,}3e-03}$	$5{,}06e-03_{3{,}8e-04}$	$5{,}22e-03_{3{,}1e-04}$
DTLZ6	$4{,}39e-02_{3{,}0e-02}$	$3{,}07e-01_{5{,}6e-02}$	$5{,}44e-03_{8{,}2e-01}$	$5{,}10e-03_{5{,}4e-04}$
DTLZ7	$1{,}03e-02_{2{,}1e-03}$	$9{,}52e-03_{2{,}1e-03}$	$5{,}03e-03_{4{,}0e-04}$	$5{,}09e-03_{4{,}7e-04}$

Tabla 8.8: Epsilon. Significancia estadística en funciones ZDT

	SPEA2					varMOPSO					SMPSO				
NSGAII	▽	▽	▲	▲	▲	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
SPEA2						▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
varMOPSO											▽	–	–	–	–

Tabla 8.9: Epsilon. Significancia estadística en funciones DTLZ

	SPEA2							varMOPSO							SMPSO						
NSGAII	▽	▽	▲	–	▽	▲	–	▽	▽	▽	▽	▽	–	▽	▽	▽	▽	▽	▽	▽	▽
SPEA2								▽	▽	▽	▽	▽	–	▽	▽	▽	▽	▽	▽	▽	▽
varMOPSO															–	▲	▲	–	▲	▽	–

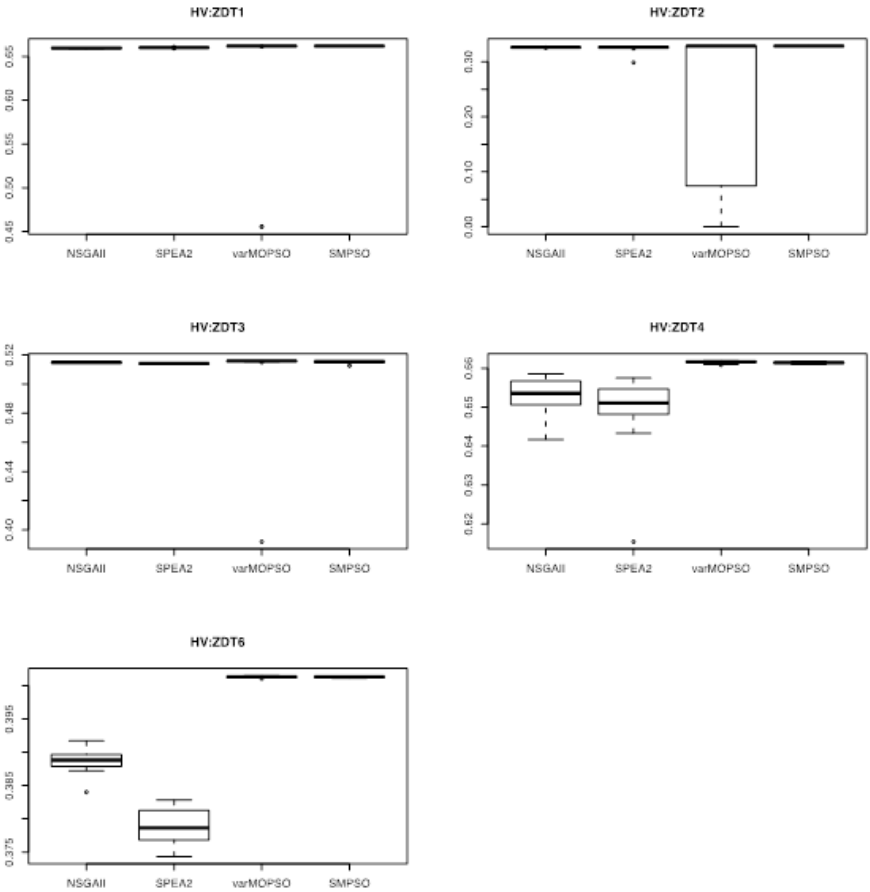


Figura 8.3: Hipervolumen. Diagrama de cajas. funciones ZDT

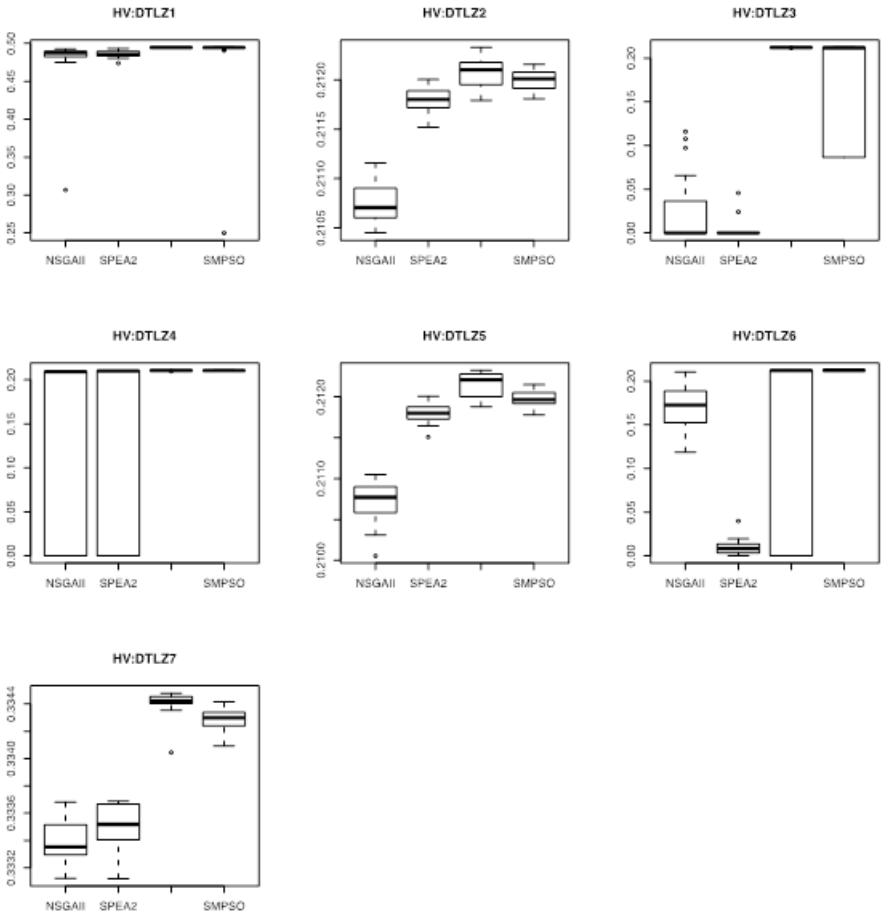


Figura 8.4: Hipervolumen. Diagrama de cajas. funciones DTLZ

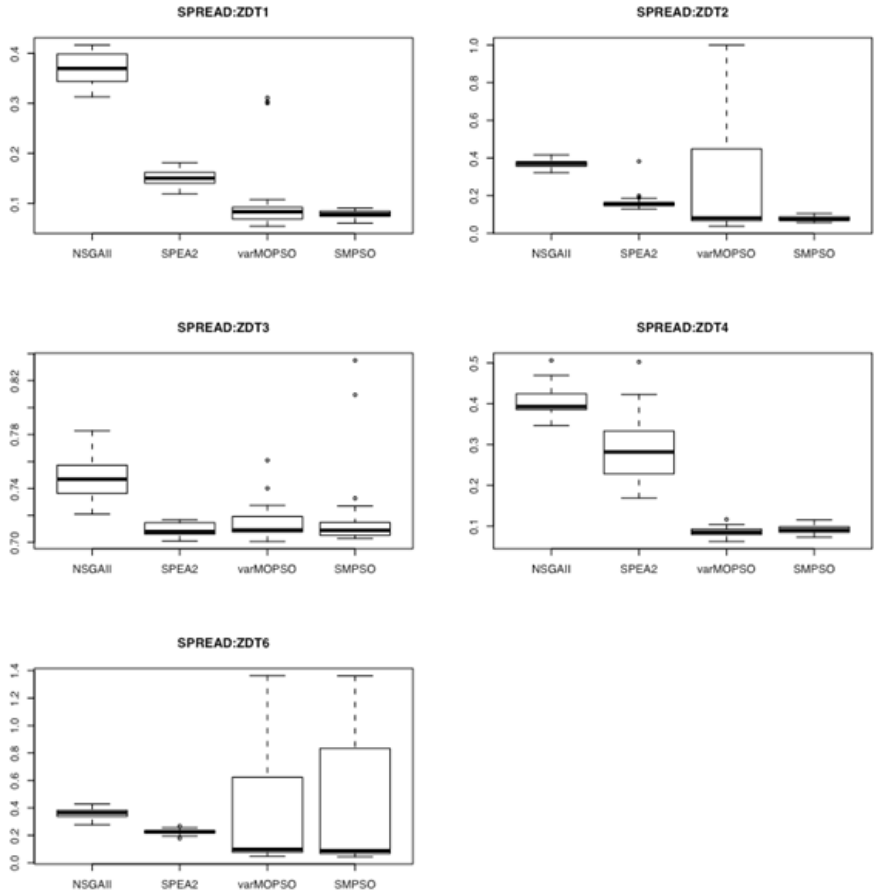


Figura 8.5: Spread. Diagrama de cajas. funciones ZDT

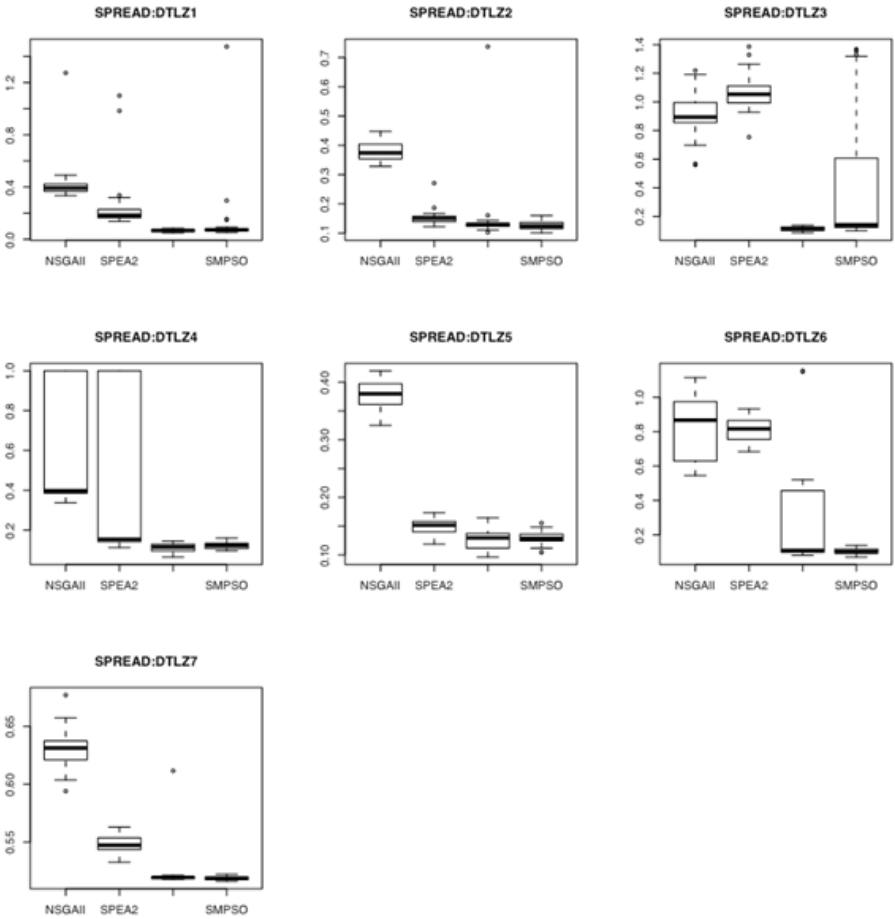


Figura 8.6: Spread. Diagrama de cajas. funciones DTLZ

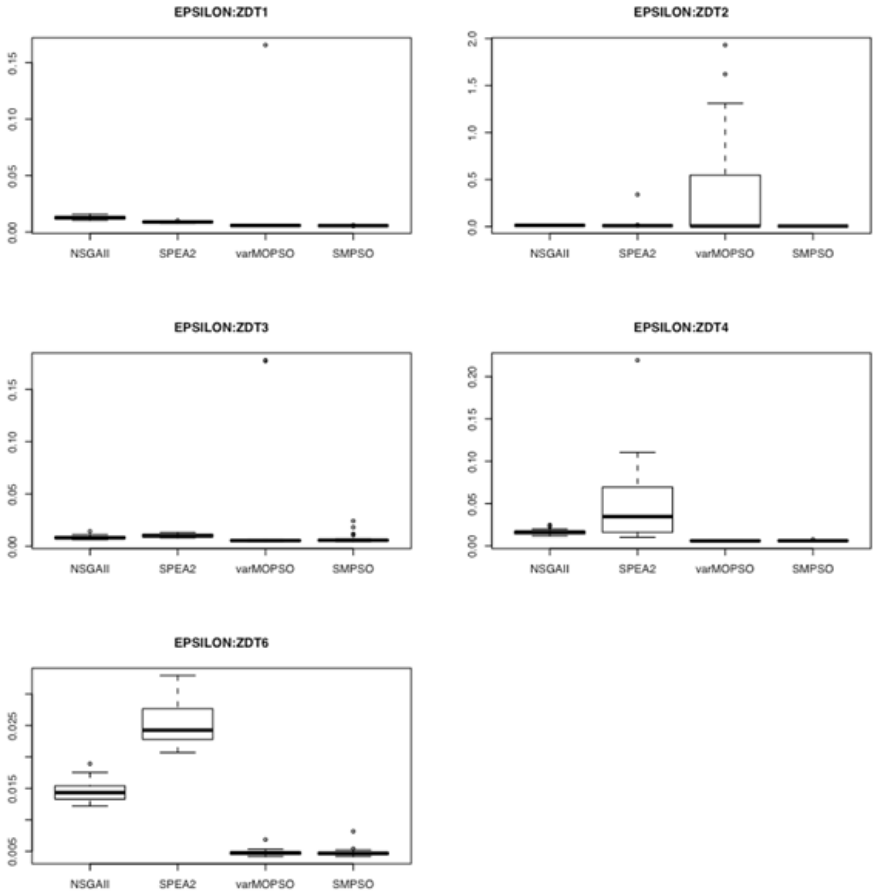


Figura 8.7: Epsilon. Diagrama de cajas. funciones ZDT

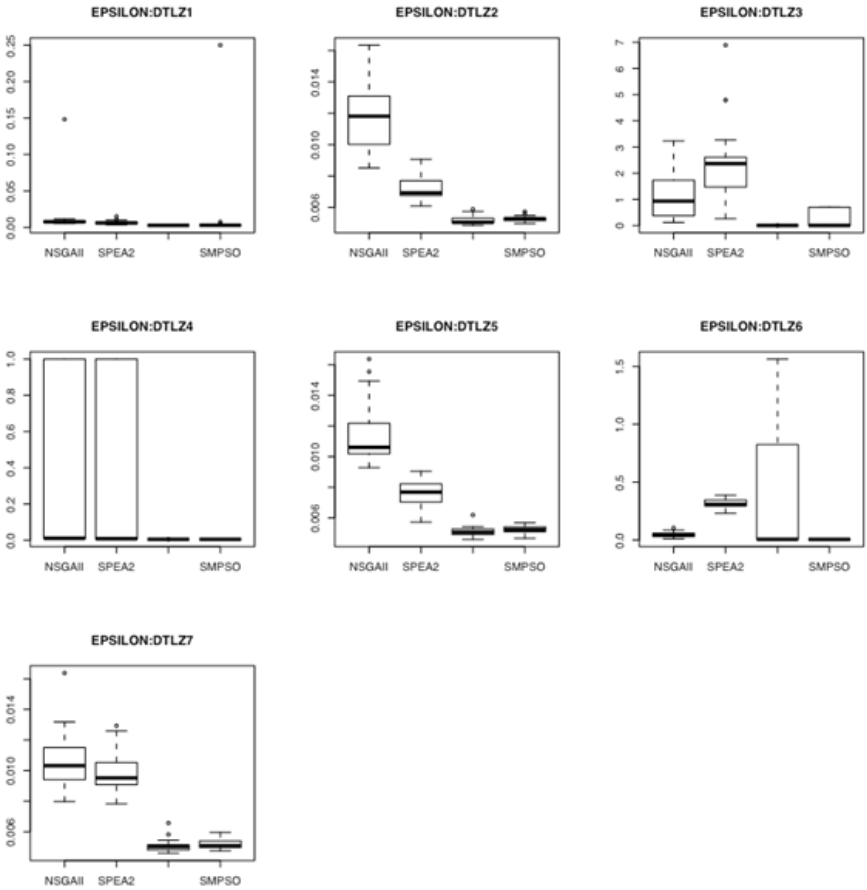


Figura 8.8: Epsilon. Diagrama de cajas. funciones DTLZ

8.4 Conclusiones

Luego de analizar los resultados presentados en las tablas mencionadas, pueden extraerse las siguientes conclusiones

- Ambos algoritmos del tipo MOPSO (SMPSO y varMOPSO) son capaces de generar frentes de Pareto de mejor calidad que los algoritmos NSGAII y SPEA2, considerando las tres métricas utilizadas y el conjunto de funciones de prueba seleccionado.

- Considerando el indicador *Hipervolumen*, varMOPSO supera el rendimiento de SMPSO en siete de las funciones de prueba, en cuatro de las funciones no se puede determinar quién presenta un rendimiento superior, y en una de las funciones SMPSO presenta un mejor desempeño.
- Considerando el indicador *Spread*, varMOPSO supera el rendimiento de SMPSO en una de las funciones de prueba, y en el resto de las once funciones no se puede determinar quién presenta un rendimiento superior.
- Considerando el indicador *Epsilon*, varMOPSO supera el rendimiento de SMPSO en tres de las funciones de prueba, en siete de las funciones no se puede determinar quién presenta un rendimiento superior, y en dos de las funciones SMPSO presenta un mejor desempeño.
- En general, varMOPSO presenta una mayor dispersión que SMPSO, pero en la mayoría de las ejecuciones el primero obtuvo el mejor resultado individual para las métricas evaluadas en las diferentes funciones de prueba.

Basado en lo presentado en este capítulo, se considera que esta propuesta es una interesante opción para resolver problemas multi-objetivo. El método está basado en la extensión a problemas multi-objetivo del algoritmo varPSO, incorporando además mejoras estudiadas en algoritmos del tipo MOPSO representativos del estado del arte.

El uso de algoritmos con tamaños de población variable es una alternativa poco explorada en el campo de optimización multi-objetivo. Cambiando el tamaño de la población, se logra una mejor adaptación al problema en ciernes, y se incrementa la posibilidad de encontrar un equilibrio entre las fases de exploración y explotación, característica deseable de todo algoritmo de optimización. Creemos, sin embargo, que existe todavía mucho trabajo que puede realizarse en el campo de MOEAs (*Multiobjective Evolutionary Algorithms*) de población variable.

En esta segunda parte se hizo hincapié en la proposición de nuevas variantes de algoritmos PSO, tanto aplicado a problemas mono-objetivo como multi-objetivo. Además, se efectuaron comparaciones de rendimiento entre las versiones propuestas y algoritmos comparables representativos del estado del arte, utilizando funciones

de prueba de laboratorio. En la siguiente parte este trabajo se focaliza en la aplicación en un problema del mundo real de los conceptos tratados.

PARTE III

Caso de Estudio

Optimización de Prestaciones Médicas

En esta tercera y última parte de la tesis se aborda la implementación de procesos de optimización basados en metaheurísticas, aplicados a una empresa del mundo real perteneciente a la industria de la salud. La empresa es Unidad Coronaria Móvil Quilmes (UCMQ), del grupo Ayuda Médica. Esta compañía es la principal empresa de emergencias y atención domiciliaria a nivel nacional. La actividad central de este tipo de empresas es la asignación de móviles a prestaciones. Para este tipo de problema, de naturaleza logística, es totalmente aplicable la utilización de algoritmos de optimización estocásticos. En la figura 9.1 se adjunta un documento comercial donde se aprecia que esta empresa constituye una de las principales compañías privadas de emergencias del país, dando cobertura a Capital Federal y Gran Buenos Aires, lo que representa un amplio porcentaje de la población total de Argentina.

Figura 9.1: Folleto Comercial Grupo Ayuda Médica

La actividad se caracteriza por requerir servicios de alta calidad, con elevados parámetros de cumplimiento, y se desarrolla a su vez en un entorno muy competitivo. El mercado exige que las compañías sean altamente eficientes, y tengan que maximizar el uso de sus recursos disponibles para lograr el cumplimiento de la demanda de servicios en los términos requeridos.

La necesidad de respuesta en tiempo real y la sensibilidad de la actividad configuran un desafío interesante para abordar con las técnicas alcanzadas por la presente tesis.

En este capítulo se presenta un revelamiento detallado del funcionamiento de esta compañía. El mismo nos permitirá entender el problema abordado desde la perspectiva de los procesos de negocio involucrados.

9.1 Introducción

La misión principal de una compañía de emergencias médicas en Argentina es proveer servicios médicos pre-hospitalarios de diferentes niveles de complejidad: emergencias médicas, urgencias, visitas médicas domiciliarias y traslado programados de pacientes a centros de atención de salud.

Estas compañías proveen servicios médicos para el inmediato y eficiente cuidado médico de los pacientes que sufren una enfermedad o sufrieron algún tipo de herida y/o lastimadura. Ellas son el segundo eslabón en la cadena de emergencia, el primero es la familia o persona que acompaña al paciente en el momento que se produce la necesidad de atención médica, y el tercero es el hospital o centro médico y sus instalaciones.

Cada incidente que se reporta a este tipo de compañías se clasifica basado en su severidad en algunas de las siguientes categorías:

1. Rojo: riesgo de muerte inminente.
2. Amarillo: emergencia seria, sin riesgo de muerte inminente.
3. Verde: visita médica domiciliaria, evento de baja criticidad.
4. Azul: traslado programado.

Como se puede apreciar, la actividad es altamente sensible dado que en las prestaciones con categoría roja o amarilla los tiempos de llegada a la resolución de una prestación pueden definir la diferencia entre la vida y la muerte. Para las prestaciones de categoría verde, el

tiempo de llegada es un factor importante, pero en ningún caso es determinante en la salud a largo plazo de la persona.

Por lo tanto, es importante hacer una primera diferenciación en cuanto a los tipos de problemáticas, las prestaciones rojas y amarillas que representan emergencias médicas y las verdes relacionadas con consultas domiciliarias. Para el caso de las emergencias médicas, el proceso de despacho (*vehicle dispatching*) se resume en enviar el móvil adecuado disponible que se encuentre más cerca del lugar de la emergencia. De esta forma, en este tipo de problemática el principal desafío consiste en lograr un nivel de cobertura (simple o doble) adecuado *previo* a la generación del incidente y así asegurar que cualquier emergencia dentro del área de servicio se resuelve dentro de los tiempos máximos establecidos. Este tipo de problema se conoce con el nombre de *Ambulance Location Problem* [234] [235] [235], existiendo amplia literatura al respecto relacionada con la definición del problema y sus posibles soluciones [237] [238] [239].

Para el caso de las prestaciones no urgentes (verdes) el problema es de naturaleza distinta, ya que no es tan importante la ubicación previa de los móviles, sino más bien el desafío principal está relacionado con contar con un proceso de despacho optimizado, que minimice los tiempos desde que la persona llama por teléfono hasta su atención médica, y que pueda brindarse con un esquema de costos sustentable para la empresa proveedora del servicio. Este problema puede considerarse como una variante del clásico *Vehicle Routing Problem (VRP)*, siendo uno de los problemas de optimización combinatoria más importantes, introducido por Dantzig and Ramser hace más de cinco décadas [a058]. Este problema pertenece al conjunto de los problemas NP-Hard [240] [241].

En los últimos años, se han utilizado diferentes metaheurísticas para abordar este tipo de problemas [239] [242] [243] [244], logrando resultados cada vez mejores, pero enfocándolos como problemas mono-objetivo. En este trabajo se explora el uso de metaheurísticas multi-objetivo para resolver el problema de un despacho óptimo para prestaciones no urgentes, minimizando los tiempos de atención y los costos del servicio en forma simultánea.

9.2 Objetivo

El objetivo del proyecto consiste en la incorporación, en la plataforma actual de software de la compañía, de una herramienta informática para automatizar el proceso de Despacho (Asignación de móviles a Prestaciones). Cabe destacar que para la compañía, las prestaciones de

categoría verde representan más del 70% de las prestaciones, y es en esta problemática donde existen mayores oportunidades de mejora.

El algoritmo será el encargado de seleccionar y asignar el móvil que debe atender cada prestación, optimizando los tiempos de llegada para la atención del paciente, aumentando la eficiencia en la utilización de los recursos disponibles, estandarizando el proceso de despacho, permitiendo definir procesos de trabajo homogéneos y con altos estándares de calidad independientemente de las personas que lo operen.

Dentro de los requerimientos iniciales de la empresa se encuentran la necesidad de optimizar dos aspectos contrapuestos, la reducción del tiempo de atención y el uso de móviles de terceros (costos). Por lo tanto, se seleccionaron metaheurísticas multi-objetivo que permiten la optimización simultánea de más de un objetivo. Este tipo de algoritmos devuelven al finalizar el proceso un conjunto de soluciones óptimas considerando todos los objetivos en juego. Este conjunto de soluciones se conoce como el frente de Pareto.

Es importante destacar la importancia de ofrecer al tomador de decisiones la posibilidad de elegir, a posteriori del proceso de optimización, entre un conjunto de soluciones óptimas del problema. De esta forma puede aprender acerca del problema, visualizar en forma gráfica la contraposición de los objetivos, analizar la interacción entre los criterios, y visualizar información estructural [150].

Una implementación exitosa de un proyecto de este tipo permite aumentar la competitividad de la empresa. Específicamente se plantea alcanzar los siguientes logros:

- Proponer un procedimiento de despacho semi-automatizado, con el propósito de optimizar y estandarizar la asignación de móviles a las prestaciones a cumplir, actuando como herramienta de soporte principal a todas las actividades de Coordinación de Cabina.

- Dar soporte a las actividades de la Dirección Médica, en la planificación de la realización del servicio, sugiriendo el cumplimiento de las prestaciones con la unidad y el equipo médico adecuados, teniendo en consideración la reducción de tiempos de llegada y la optimización de los recursos disponibles.

- Brindar información acerca del estado del sistema, tiempos de demoras estimados, resumiendo en gráficos de fácil lectura el

cuadro de situación general, permitiendo tomar acciones correctivas por parte de la Dirección Médica en caso de considerarse necesario.

- Agregar el soporte de cartografía digital, en la toma de la prestación para cálculo de ubicación exacta, así como en las pantallas que muestran el estado de situación.
- Incorporar a la aplicación de Administración de Cabina herramientas para la estandarización de las tareas de seguimiento y control de las prestaciones y móviles.

9.3 Relevamiento de Situación Actual

En este capítulo se describen con grado de detalle los procesos de trabajo actuales relacionados con la recepción de llamados, asignación del despacho, seguimiento y resolución de los mismos. Estos procesos comprenden todas las actividades desarrolladas dentro del ámbito de la Cabina de la UCMQ. Se analiza además aspectos puntuales de la aplicación informática que brinda soporte a estas tareas.

Fuentes de Información

Esta documentación se generó a partir de las siguientes fuentes de datos:

1. Relevamientos
   (a) Responsable de Sistemas de Unidad Coronaria Móvil Quilmes
   (b) Responsable de Coordinación de Cabina de Unidad Coronaria Móvil Quilmes
2. Información general provista por la compañía
3. Trabajo de campo en UCMQ
4. Presentación en reuniones de Directorio
5. Normas ISO-9000:
   (a) Manual de Calidad Grupo UCMQ
   (b) Anexo I - Diagramas de Manual

(c) Normas para Radio-Operadores

(d) Formación Profesional

(e) Coordinación Médica

(f) Área Técnica

(g) Recepción Despacho y Asist. Médica

9.4 Descripción del Proceso

Es esta parte del trabajo se detallan las actividades que se encuentran bajo la responsabilidad del Coordinador de Cabina, y que tienen relación directa con la resolución de prestaciones médicas.

Las podemos agrupar en las siguientes categorías

- Actividades de planificación y administración de recursos
    - Solicitud y seguimiento de recursos operativos (Guardias)
    - Disponibilidad de móviles (recursos operativos) para atender las prestaciones
    - Recursos humanos en la cabina
- Actividades operativas vinculadas a la resolución de prestaciones
    - Recepción de prestaciones
    - Asignación de prestaciones
    - Seguimiento de prestaciones

En ningún caso se releva ni se analiza las actividades vinculadas con el acto médico en sí, en el momento de atención al paciente o en momentos previos en forma de instrucciones de pre-arribo o similar.

Figura 9.2: Equipo médico habitual - Unidad de Terapia Intensiva Móvil - UTIM

Actividades de Planificación y Administración de Recursos

En esta sección se describen los procesos relacionados con el armado de los equipos en cada móvil, turnos y guardias médicas.

Respecto de la solicitud y seguimiento de los recursos operativos

1. En base a la demanda de prestaciones, considerando demanda estimada, información histórica y en base empírica el Coordinador de Cabina efectúa el pedido de cantidad de recursos operativos (móviles de guardias) al área de Coordinación Médica. Esta negociación es permanente, y se monitorea en forma diaria por parte de los responsables del área.

2. El área de Coordinación Médica informa al Coordinador de Cabina la cantidad de recursos operativos (móviles de guardias), así como los médicos y/o choferes que componen cada uno de los recursos disponibles. Esta información se envía por correo electrónico.

3. El Coordinador de Cabina confirma que las guardias comiencen de acuerdo a lo planificado. Registra en el sistema los recursos operativos con los que cuenta. En una planilla de cálculo se

registra la hora de inicio y fin de la guardia y deja registro de las novedades del día. Esta información no se vuelca al Sistema de Administración de Cabina, solo se registra para comunicar por correo electrónico.

4. El Coordinador de Cabina informa al área de Coordinación Médica las guardias que no comenzaron de acuerdo a lo previsto, y cualquier otra novedad que considere adecuado.

Respecto de la disponibilidad de móviles (recursos operativos) para atender las prestaciones

1. Se cuenta con una disponibilidad de 80 UTIM´s (Unidad de Terapia Intensiva Móvil), ubicadas en bases pre-fijadas.
2. Se cuenta con un número aproximado de 130 móviles estándar (auto con médico) propios, algunos de ellos con médicos en relación de dependencia y otros en modalidad contratado.
3. Se cuenta con un número aproximado de 50 móviles externos. Estos móviles pertenecen a otras empresas que brindan el servicio en modalidad tercerizados.
4. Las guardias de UTIM empiezan y terminan siempre en los mismos horarios.
5. Las guardias de móviles estándar empiezan y terminan en diferentes horarios programados.
6. Existe una unidad UTIM por base. Las UTIM´s en general, vuelven a la base una vez cerrada cada prestación. En la figura 9.2 se puede apreciar una Unidad de Terapia Intensiva Móvil.
7. Las unidades móviles estándar no regresan a la base entre prestación y prestación.
8. El médico visitador (no UTIM) está en condiciones de resolver prestaciones categorizadas como verdes y amarillas.
9. Las bases son fijas. En verano, se dan de baja 2 bases por falta de demanda.
10. El horario de almuerzo lo determina la cabina para el caso de las UTIM. El mismo se interrumpe, en caso de prestaciones categorizadas como amarillo o rojo.

11. Hay médicos de móvil estándar que no hacen prestaciones categorizadas con el color rojo, o no hacen pediatría, o no hacen gerontología.

12. Cada móvil realiza un mínimo de prestaciones. Esto se hace de esta forma para lograr una utilización uniforme de los recursos y un esquema rentable por unidad.

13. La tabla 9.1 muestra los lugares donde se cubren en forma directa las prestaciones. Son 14 partidos pertenecientes al Gran Buenos Aires (GBA) además de la Ciudad Autónoma de Bs.As.

14. Hay zonas que se atienden exclusivamente con terceros, que se corresponden con la zona del GBA y La Plata no comprendido en el párrafo anterior.

15. En forma directa e indirecta se cubren los 24 partidos pertenecientes a los dos primeros grupos de partidos de la provincia de Buenos Aires que rodean la ciudad autónoma, totalizando un área de $3.833 km^2$. Esta área tiene una población estimada de 12.800.000 personas (ver figura 9.3).

16. Existe además recursos externos cubriendo las mismas zonas que los móviles propios, dado que la demanda promedio es mayor a la que se le puede dar servicio. Con estos proveedores se trabaja en forma regular, y se le asegura un piso de servicios para que de esa forma pueda planificar los recursos que pone a disposición.

17. Hay zonas rojas, donde se acude con un móvil del tipo UTIM, independiente del código de categoría (a veces con acompañamiento policial). Estas zonas no están identificadas formalmente, y forma parte del conocimiento y experiencia del despachador.

Lugares de prestación del servicio

Ciudad de Buenos Aires	Avellaneda
General San Martín	Tigre
Hurlingham	Ituzaingó
José C. Paz	Lanús
Lomas de Zamora	Malvinas Argentinas
Morón	Quilmes
San Isidro	San Miguel
Tres de Febrero	Vicente López
Almirante Brown	Berazategui
Esteban Echeverría	Ezeiza
Florencio Varela	La Matanza
Merlo	Moreno
San Fernando	

Tabla 9.1: Partidos del Gran Buenos Aires donde se cubren las prestaciones

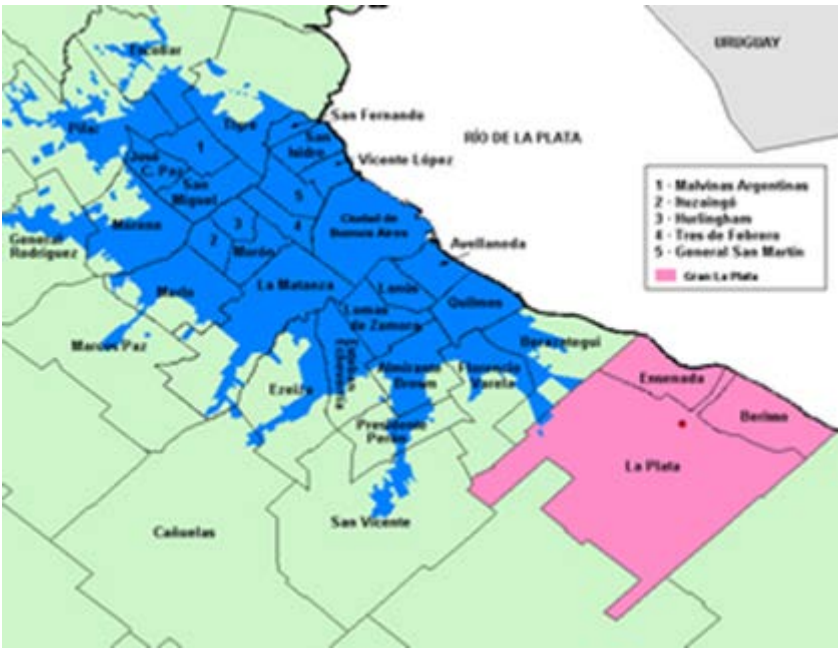


Figura 9.3: Ciudad de Buenos Aires y partidos del GBA donde se presta el servicio

Respecto de los recursos humanos en la cabina

1. Hay rotación de operadores y despachadores en la cabina, entre 2 y 4 por año.
2. Los turnos en la cabina están definidos para que funcionen en forma uniforme, intercalando recursos que concurren días laborables y durante fines de semana y feriados. De esta forma, se elimina el efecto que la cabina funcione en diferentes modalidades de lunes a viernes y los fines de semana.
3. El personal de cabina, despachadores y operadores manifiestan que el apoyo geográfico es central, con utilización de cartografía.

Actividades Operativas Vinculadas a la Resolución de Prestaciones

A continuación se describe el proceso de recepción, despacho y cumplimiento de la prestación, detallando solamente aquellos aspectos relevantes para la optimizacion de prestaciones médicas.

Recepción de Prestaciones

Todas las prestaciones se reciben en la cabina e ingresan por las siguientes vías:

1. Teléfono, directamente desde el lugar donde se genera la prestación.
2. Internet, ingresando directamente al Sistema de Administración de Cabina.
3. Teléfono, desde otras cabinas tomadoras.

Al ingresar las mismas se registra la siguiente información:

- Fecha y hora
- Código de categoría de prestación (Color tomado)
    - Rojo ’ emergencia, con riesgo de vida
    - Amarillo ’ urgencia
    - Verde ’ no urgencia

    - Celeste ’ verde con prioridad
    - Azul ’ traslado programado
- Nombre y apellido del paciente.
- Domicilio - Localidad - Partido.
- Despacho
    - Principal (rojos y amarillos)
    - Eje Zona Sur
    - Eje Zona Oeste
    - Eje Zona Norte
    - Capital Federal A
    - Capital Federal B
    - Capital Federal C
    - Capital Federal D
    - Capital Federal E
    - Externos (Terceros que brindan servicios)
- Tomador
- Edad (adulto - infante)
- Unidad de negocio (convenio)

El código de categoría de prestación se define de acuerdo a una categorización establecida en el protocolo de recepción de llamadas. Se efectúan una serie de preguntas estándar (Triash), y de acuerdo a las respuestas recibidas se define el código de categoría.

El Triash del sistema sirve como soporte acerca de las preguntas a realizar al paciente o a la persona que está llamando. No se registran las respuestas de la persona que llama. En caso de existir información significativa, la misma se ingresa como una nota no estructurada en el Sistema de Administración de Cabina.

El proceso de asignación de categoría (rojo, verde, amarillo) es crucial. En algunos casos, se cambia la categoría producto de la

demora. A veces, las pre-pagas lo informan con otro código, motivado en que no lo reportaron a tiempo.

Estas prestaciones ingresan a una cola de prestaciones a servir (Despachos). Cada prestación pertenece a un único despacho, en un único momento. Cada prestación se asigna a un despacho en función de la categoría, zona y tipo de recurso. Esta asignación por defecto ofrecida por el Sistema de Administración de Cabina se puede modificar, y por lo tanto esa prestación cambia de despacho. La información de prestaciones generalmente se visualiza agrupada por despacho. Esta división por despacho es arbitraria, y es una forma de balancear la carga de trabajo entre los despachadores. Las prestaciones con código de categoría rojo se visualizan en todos los despachos en todo momento. De esta forma, se minimizan los riesgos de falta de asignación a prestaciones críticas. Los despachos tienden a ser compartimientos estancos.

En todo momento se puede ver la situación de una prestación, identificando los siguientes estados:

Estado	Descripción
Inicial	Sin visualizar por ningún despachador
Sin asignar	Sin asignación de recurso operativo (móvil)
Asignada pendiente	Con asignación de móvil, que aún no llego al domicilio
En camino	Recurso en camino a una prestación en particular (opcional)
En domicilio	Recurso operativo (móvil) arribado a domicilio
En Derivación	Prestación que generó un traslado del paciente a un centro de internación
Cerrada	Prestación concluida y móvil liberado de esa prestación

Asimismo, se visualiza el tiempo que transcurrió desde el llamado inicial, hasta la fecha/hora actual o la hora de llegada al domicilio. Una vez "En domicilio", se registra el tiempo que lleva en ejecución la prestación. La información de estados se mantiene actualizada en tiempo real, por la utilización del sistema WAP-Nextel.

La figura 9.4 muestra un diagrama de transición de estados en lo que puede encontrarse una unidad móvil, sea esta una Unidad de Terapia Intensiva Móvil (UTIM) o un móvil de baja complejidad.

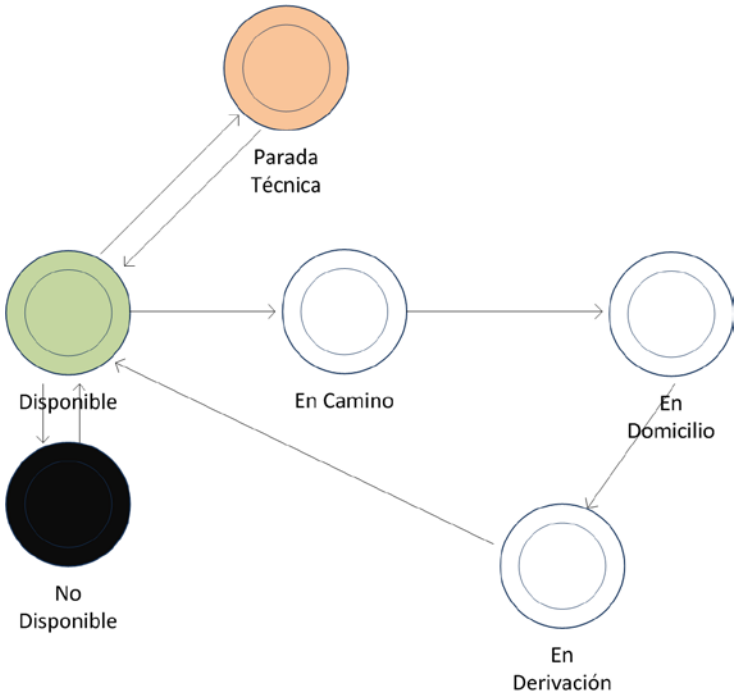


Figura 9.4: Diagrama de Transición de Estados de un recurso operativo (Móvil)

En la figura 9.5 se puede visualizar los estados por lo que puede pasar una prestación, cualquiera sea la categoría de la misma.

Figura 9.5: Diagrama de Transición de estados de una prestación

El tiempo máximo de llegada que se considera como referencia es de 15 minutos para las prestaciones categorizadas como rojo, aunque en la realidad se intenta solucionarlo tan pronto como se pueda. El tiempo máximo de llegada que se considera como referencia para los amarillos es 30 minutos. Se resuelven entre este periodo de tiempo y 1 hora. Para los verdes, el tiempo es 3 o 4 horas, hasta 10 horas, como lapso de cumplimiento estándar. Con el servicio saturado, pueden ser hasta 72 horas. El grupo Ayuda Médica realiza muy pocos traslados programados. Los mismos se realizan sin médico.

El cálculo de ubicación geográfica de un recurso operativo (móvil) se efectúa a partir de la última ubicación reportada, y un cálculo empírico. A veces, se chequea vía teléfono. Los móviles tipo UTIM poseen un dispositivo de posicionamiento satélital (GPS). Accediendo al software del mismo, se puede ver la ubicación del móvil en tiempo real. No existe ninguna integración entre el Sistema de Administración de Cabina y el software de GPS.

Hay que balancear el uso de los móviles para que resulte un esquema rentable y sustentable en el tiempo. El promedio de prestaciones cerradas es entre 2 y 3 prestaciones por hora por cada móvil.

En caso de que exista un móvil del tipo UTIM sin asignación durante más de 1 hora, el mismo se utiliza para la resolución de códigos verdes en zonas cercanas a la base. Se interrumpe esa asignación en caso de surgir prestaciones con código rojo.

El sistema muestra el historial de uso de un móvil, ingresando a la pantalla detallada de uso de ese recurso operativo.

Los códigos verdes se priorizan, considerando si es afiliado directo, o dependiendo de la empresa pre-paga del paciente y/o su plan de salud. Las prestaciones categorizadas como códigos amarillos que se encuentran demoradas, pueden transformarse a rojo si varía el cuadro médico. En aquellas prestaciones categorizadas con código rojo o amarillo, se llama al lugar de la prestación para brindar instrucciones de pre-arribo. Este procedimiento es de central importancia para resolver la prestación de una forma exitosa.

Asignación de prestación (Despacho)

Cada despacho tiene un despachador responsable. El despachador es el encargado de asociar un recurso (UTIM, móvil estándar, tercero, ambulancia para traslado) a una prestación. Una vez realizada esta asociación, la misma es comunicada vía WAP-Nextel a la unidad.

Al momento de asignar el despacho, se define una categoría (rojo, amarillo o verde). Esta categoría no tiene que ser la misma que la

categoría de la prestación, aunque coincide para la mayoría de los casos. Al recurso operativo (médico) se le informa la categoría de la asignación del despacho.

El médico de la unidad estándar de visita (no UTIM) puede rechazar la prestación en caso que considere que no esta capacitado a brindar la misma, excepto el caso que sea un código rojo.

Las prestaciones con código rojo pendiente de asignación son visualizadas en todos los despachos en un sector destacado, primeros en la lista. Para códigos amarillo o rojo, siempre se asigna la unidad UTIM más cercana. De no existir una cerca, se puede cubrir temporalmente con una unidad móvil estándar (no UTIM). De no existir demanda, es posible utilizar una UTIM para una prestación diferente de amarillo o rojo. A los móviles estándar y UTIM se despachan entre 3 y 4 asignaciones, para evitar que la lista de pendientes se incremente en exceso. El médico responsable del móvil puede alterar el orden de las prestaciones a atender, respetando la priorización natural por categoría (rojo, amarillo y verde). En caso de prestaciones priorizadas, se informan al médico formalmente por teléfono. No se priorizan en el Sistema de Administración de Cabina. No se cambia el orden de prestaciones en el sistema WAP-Nextel. Las prestaciones con reclamos tienen precedencia sobre las no reclamadas, a igual categoría. Tiempo atrás, las prestaciones se pasaban de a 1 a los médicos. No se asignaba una nueva hasta no cerrar la prestación actual. Esto permite se mantenga el orden de asignación establecido en cabina, pero el médico no sabe su carga de trabajo restante. Este procedimiento se utiliza ocasionalmente para corregir malos hábitos, en casos donde los médicos no informan vía WAP-Nextel el estado de la prestación. En el sistema, se pueden identificar qué móviles son multiasignables, o si sólo pueden tener una única asignación. Existen re-asignaciones en forma permanente en función de la demanda en cada momento. Estas re-asignaciones se informan por WAP-Nextel y se confirman por teléfono. Existen casos donde se categoriza mal el código de prestación, por temas de criterio o mala información brindada por el paciente. No se hacen seguimiento de estos casos, comparando los síntomas reportados por el paciente con el diagnostico brindado por el médico. La cantidad de re-categorizaciones es un indicador de la experiencia del Operador. Operadores con poca experiencia, pueden realizar un mal diagnostico de la situación y categorizar o re-categorizar en forma errónea.

Las empresas externas donde se tercerizan servicios tienen un dispositivo WAP-Nextel, y todos los móviles externos se visualizan en el sistema como uno. Las prestaciones que se les asignan se informan vía WAP-Nextel. Las empresas terceras informan el estado y el cierre

de las prestaciones vía WAP-Nextel. Es posible cambiar el estado de más de una prestación por vez, lo que posibilita la registración del estado correcto en estos casos. Se debe validar que este procedimiento sea estándar para todos los casos. Los externos resuelven prestaciones categorizadas como rojo, amarillos o verdes, dependiendo del tipo de recurso operativo que poseen, UTIM y/o móvil estándar. En el sistema de Administración de Cabina no se registra el tipo de recurso operativo con el que dispone el tercero. Todos los externos se administran como un despacho independiente, sin agrupar por zona geográfica.

Seguimiento de las prestaciones

El despachador debe realizar un seguimiento de la prestación, analizando en forma permanente las prestaciones que están pendientes de asignar, así como las prestaciones asignadas pendientes de resolución. También debe realizar un seguimiento de las prestaciones que se encuentran en proceso de ejecución (En domicilio), llamando cada 10 minutos una vez iniciada la misma. Esta tarea no se registra en el Sistema de Administración de Cabina. El sistema de Administración de Cabina no identifica las prestaciones en ejecución con más de 10 minutos que no tienen llamados de seguimiento.

En el caso de prestaciones categorizadas como rojas o amarillas, el despachador llama al domicilio del paciente para dar instrucciones de pre-arribo, y monitorear el estado de salud del mismo. No se registra en el sistema si se ha efectuado o no el llamado de pre-arribo. Por lo tanto, el sistema no identifica las prestaciones urgentes que aún no tuvieron llamado de pre-arribo. La información relevante que surja del llamado de pre-arribo se puede documentar en el Sistema de Administración de Cabina como nota no estructurada en la prestación. La información de Notas no se envía por el sistema WAP-Nextel.

En forma manual, se actualiza regularmente la ubicación de cada móvil. Es de utilidad, en caso de traslados o derivaciones que requieren un tiempo prolongado.

El seguimiento de prestaciones se efectúa en base a la experiencia del despachador, el Sistema de Administración de Cabina no identifica información relacionada con el seguimiento.

A continuación se detallan los eventos centrales relacionados con el seguimiento de una prestación:

1. Reclamos: Los reclamos recibidos, en general asociados a demoras en los tiempos, se registran en el sistema como notas asociadas a la prestación. De esa forma, el despachador puede

ver on-line aquellas prestaciones que tiene reclamos asociados. Las prestaciones aún no resueltas que fueron reclamadas, se priorizan. El tiempo de demora estimado se calcula en forma empírica.

2. Cumplimiento de la Prestación: En este párrafo se describen las acciones vinculadas al cumplimiento de la prestación, en referencia a los procesos vinculados con el sistema de información. No se detallan ni analizan los procesos vinculados con el acto médico.

   Cuando se llega al domicilio de la prestación, esta información se actualiza en forma on-line utilizando el sistema Wap-Nextel (Estado En domicilio). Transcurridos 10 minutos desde la llegada al domicilio, el despachador tiene que efectuar el seguimiento de la prestación llamando al móvil correspondiente.
3. Derivación: Los códigos rojo y amarillo pueden generar derivación a un instituto médico. La misma se efectúa generalmente con el mismo móvil, aunque a veces la derivación se puede realizar con otro móvil. La derivación debería ser sólo dentro del Gran Buenos Aires en prestaciones de GBA, aunque existen casos que se efectúan derivaciones a Capital Federal. La misma situación se da al inversa, para las prestaciones de Capital Federal. En caso de corresponder una derivación, el despachador será el responsable de gestionar la misma. Una vez acordada la derivación con la institución médica receptora del paciente, el despachador le suministra toda la información necesaria al móvil para que realice la derivación al lugar que corresponda. En una derivación se indica hacia dónde va (localidad) en el Sistema de Administración de Cabina. El domicilio o institución médica donde se deriva se ingresa como nota no estructurada. La derivación es un caso particular de traslado. Para los traslados, se informa en el sistema los diferentes tramos del traslado. Esta información queda asociada a la prestación.
4. Fin de la Prestación: Cuando se finaliza la prestación, esta información se actualiza en forma on-line utilizando el sistema WAP-Nextel (Prestación finalizada).

Para cerrar una prestación se debe registrar:

(a) Diagnostico - obligatorio

(b) Co-pago, de existir - opcional

(c) Medicación utilizada - opcional

El diagnóstico y la medicación se pueden ingresar vía sistema Wap-Nextel por parte del médico. En caso que el médico no lo realice, lo ingresa el despachador. El inicio y fin de la guardia no se registra vía WAP-Nextel, dado que puede existir un problema de acceso al teléfono Nextel entre los diferentes integrantes del equipo médico.

En este capítulo se presentó un relevamiento del funcionamiento de la empresa Unidad Coronaria Móvil Quilmes, en los que respecta a los procesos vinculados con la resolución de prestaciones médicas. El mismo nos sirve para lograr un entendimiento del funcionamiento general de la compañía, así como brindar el marco de referencia para entender en detalle el problema de optimización que deben resolver este tipo de compañías en forma diaria.

Propuesta de Solución

En este capítulo se describe en detalle la propuesta de solución ofrecida a la compañía dentro del ámbito de este proyecto. Esta propuesta considera todos los desarrollos, modificaciones y adquisiciones de software previos y posteriores que permiten implementar el proceso de optimización del despacho automatizado. El enfoque de este proyecto es del tipo integral, y la solución cubre todas las tareas necesarias para alcanzar los objetivos planteados. El módulo central del presente trabajo es la automatización del proceso de asignación de móviles a prestaciones médicas, llamado Asignación Automática. Este proceso se basa en la utilización de algoritmos evolutivos multi-objetivo. Entre los módulos más destacados complementarios a la solución algorítmica propiamente dicha podemos mencionar la aplicación de Cartografía Digital (GIS - *Geographic Information System*). Esta aplicación permite definir las coordenadas a partir de una dirección (geo-localizar) y provee todos los mapas y las APIs (Interfaces de Aplicación de Programas) para mostrar la información geo-referenciada.

También podemos mencionar el módulo de Seguimiento Automatizado, con el objetivo de estandarizar los procesos de trabajo y asegurar el correcto cumplimiento del servicio.

Por las características de la solución ofrecida, y dado que el algoritmo de optimización se embebe dentro del Software de Cabina existente en la compañía, el desarrollo y ajustes de interfaces es un punto destacado dentro del plan de tareas de este proyecto.

En el presente capítulo vamos a realizar, en primera instancia, una descripción general resaltando las características de la solución. A continuación, se realizará un descripción del proceso propuesto y la utilización de las nuevas herramientas de software. Para finalizar, se describen en detalle los tres componentes principales, la cartografía digital que permite ubicar en un plano todos los elementos en juego, el proceso de asignación automática que posibilita generar una asignación de móviles a prestaciones optimizada y el módulo de seguimiento cuyo objetivo consiste en mejorar la calidad del servicio prestado.

10.1 Beneficios

En esta sección se describen los principales beneficios asociados con la solución propuesta:

- Asignación de recursos en forma optimizada, reduciendo tiempos de llegada a las diferentes prestaciones, aumentando la calidad de servicio y maximizando el rendimiento de los recursos disponibles.
- Mantenimiento de estado general del sistema, que resuma estado de situación en cada momento, permitiendo tomar acciones correctivas y preventivas de acuerdo a criterio de la Dirección Médica.
- Generación de información estimada de arribo a las diferentes prestaciones a cumplir, de acuerdo al estado del sistema, información histórica de servicios, y simulación de uso de recursos.
- Aumento de la eficiencia en la utilización de recursos.
- Incremento de la información disponible para los despachadores, dada la utilización de cartografía digital base.
- Estandarización del proceso de despacho, considerando el estado de situación completa.

10.2 Consideraciones

En esta sección se listan las puntos más relevantes a considerar:

- Siempre que sea posible se obtiene la ubicación exacta de los elementos involucrados en el proceso de despacho (prestaciones, móviles, bases o centros de internación). Cuando no se cuente con información exacta, se prevé utilizar como criterio de asignación y tiempo de llegada un diagrama de distancias entre las diferentes localidades del conurbano. Estas distancias se calcularán en base a los baricentros de las localidades dentro del área de cobertura, y los tiempos se calcularán en base a promedios históricos.
- Es necesario la registración de los tiempos de no disponibilidad de los recursos.
- Es necesaria la registración de los tiempos de inicio y fin de servicios de guardias, para asignar de forma correcta.

- Es necesario identificar por cada recurso que brinda servicio, qué tipo de prestaciones puede cubrir.
- El software a implementar actúa como herramienta de soporte al despachador, proponiendo la asignación del servicio en base a minimización de tiempos, recursos disponibles, código y tipo de la prestación, y sujeto a todas las restricciones inherentes de los recursos posibles. El despachador podrá aceptar y/o modificar la asignación propuesta.
- La información de estado de situación general, así como de tiempo de respuesta estimado para cada prestación, es realizada en base a la utilización de información estadística disponible en las bases de datos de la compañía.
- Toda información de tiempos de respuesta promedio es acompañada por una medida de variabilidad (varianza) a los efectos de visualizar y entender el grado de exactitud de la estimación realizada.
- La herramienta a desarrollar está completamente integrada a la plataforma existente, y supone la colaboración y participación directa del área de sistemas de la empresa.

10.3 Características del Proyecto

En esta sección se define en detalle las características centrales del proyecto. Se detallan y describen los tres módulos principales que conforman la solución propuesta:

1. Asignación Automática
2. Cartografía Digital
3. Seguimiento Automatizado

Asimismo, se identifican las interrelaciones entre los mismos, y la integración necesaria entre estos tres componentes y el software principal de la Cabina.

Especificación General

La figura 10.1 muestra un diagrama de alto nivel, donde es posible visualizar los tres módulos de software nuevos que se agregarán al sistema actual de Administración de Cabina:

- Asignación Automática
- Cartografía Digital
- Seguimiento Automatizado de Prestaciones

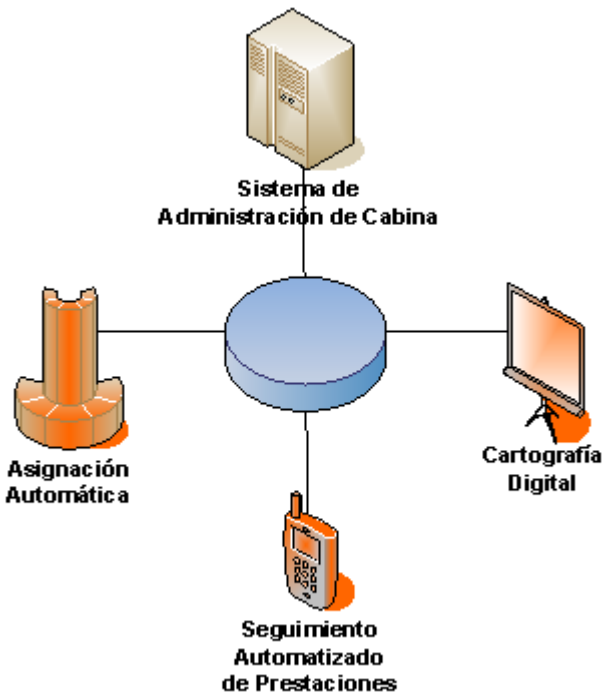


Figura 10.1: Diagrama de alto nivel de la solución

El módulo de **Asignación Automática** será el responsable de seleccionar en tiempo real el recurso que atenderá cada prestación. Los criterios de asignación serán definidos a nivel Dirección.

El módulo de **Cartografía Digital** tendrá como misión brindar la funcionalidad de mostrar información sobre mapas digitales, tanto a los efectos de validar información como de mostrar el estado de las prestaciones.

El módulo de **Seguimiento Automatizado de Prestaciones** tiene como objetivo estandarizar todas las acciones vinculadas con el seguimiento de las prestaciones y móviles, procurando que todos los despachadores trabajen de una manera uniforme, y que las políticas de seguimiento de las prestaciones se definan a nivel de Dirección.

10.4 Descripción del Proceso Propuesto

A continuación se detallan las tareas principales del proceso, detallando los agregados y/o cambios a realizar con el propósito de cumplir los objetivos del presente proyecto.

Toma de la prestación

Durante el proceso de toma de llamado **se agregará el soporte de cartografía digital**. Este agregado nos permitirá por un lado, **validar que la ubicación reportada por la persona que realiza el llamado sea correcta** (certificando calle, altura, intersecciones, localidad, etc.) visualizando un mapa y el domicilio proyectado en el mismo y por otro lado obteniendo las coordenadas exactas (longitud y latitud) del domicilio validado. Estas coordenadas serán utilizadas en el momento de asignación de móviles a prestaciones.

Prestación recibida en cabina

En las prestaciones recibidas en cabina en forma telefónica, la tarea anteriormente descrita se realizará mientras transcurre el llamado telefónico asociado a la prestación. En caso de no encontrar el domicilio por calle, numeración y localidad se permitirá la opción de buscarlo en forma manual en el mapa. En caso de no poder ubicar el domicilio en forma exacta, se registrará de forma similar que en la actualidad.

Prestación recibida por Internet

Existirá un responsable de cabina cuya tarea será la de obtener las coordenadas exactas de las prestaciones que llegan vía Internet. La aplicación contará con un módulo para tal fin, donde se considerará minimizar el trabajo manual para la obtención de las mismas. En caso de no poder ubicar el domicilio en forma exacta, se registrará al igual que en la actualidad asegurando la correspondencia

de la localidad con la información almacenada en el Sistema de Administración de Cabina. Se evaluará la posibilidad de realizar una integración entre las tablas de las aplicaciones de las diferentes cabinas, para minimizar la información a completar en forma manual.

Despacho

Este proceso es el que sufrirá los cambios más significativos. **La aplicación elegirá en forma automática el despachado/despachador que atenderá la prestación**.

Utilizará como criterio de asignación la afinidad entre los despachos/despachadores y la zona geográfica y la carga de trabajo acumulada en cada despachador. El despachador podrá rechazar la asignación indicando un motivo. Para estos casos, el Coordinador de Cabina deberá asociar un despachador a esas prestaciones en forma manual.

La aplicación asignará en forma automática cual es el móvil que atenderá cada prestación. El despachador confirmará o rechazará esta asignación. En caso de rechazo, el despachador deberá indicar un motivo de no aceptación tabulado en la aplicación, y asignar un móvil en forma manual. (Ver párrafo **Asignación Automática**)

La aplicación mostrará en todo momento las prestaciones sin asignar, las asignadas automáticamente sin aceptación del despachador, y las asignadas y confirmadas por el despachador.

Prestaciones sin asignar

Estas prestaciones se visualizarán en la modalidad actual (pantalla del sistema). Adicionalmente, se visualizarán en un mapa digital filtrando por diferentes criterios (despachador, localidad, tiempo de demora).

Asignación de móviles a prestaciones

La aplicación efectuará una relación prestación-móvil en forma automática. Dependiendo de la categorización de la prestación (rojo/amarillo/verde) los criterios de asignación serán diferentes.

1. Rojo **El algoritmo de asignación automática a implementar en el sistema de Administración de Cabina asignará el móvil UTIM más cercano al lugar de la prestación**. La aplicación le mostrará al despachador un mapa digital con el lugar de la prestación, los 4 móviles más cercanos identificando tipo de móvil (UTIM, móvil estándar) y estado (disponible, en camino, en domicilio, parada técnica, etc.) **Si la aplicación**

detecta que el tiempo de llegada estimado de la UTIM más cercana libre excede el tiempo mínimo para esta categoría de prestación (15 min.) y si existiese un móvil estándar a una distancia menor al tiempo mínimo, asignará esta prestación a ambos móviles, identificando este caso particular. Esta asignación permitirá que se efectúen algún tipo de acción de emergencias mientras se espera la llegada del móvil UTIM. En todos los casos, el despachador realizará los llamados, validaciones, y controles que el protocolo indique en estas situaciones.

2. Amarillo **El sistema de Administración de Cabina asignará el móvil UTIM más cercano al lugar de la prestación**. La aplicación le mostrará al despachador un mapa digital con el lugar de la prestación, los 4 móviles más cercanos identificando tipo de móvil (UTIM, móvil estándar) y estado (disponible, en camino, en domicilio, parada técnica, etc.). **Si la aplicación detecta que el tiempo de llegada estimado de la UTIM más cercana libre excede el tiempo mínimo para esta categoría de prestación (30 min.) y si existiese un móvil estándar a una distancia menor al tiempo mínimo, asignará esta prestación a ambos móviles, identificando este caso particular**. Esta asignación permitirá que se efectúe algún tipo de acción de emergencias mientras se espera la llegada del móvil UTIM. En todos los casos, el despachador realizará los llamados, validaciones, y controles que el protocolo indique en estas situaciones.

3. Verde Estas asignaciones constituyen el 75% de la cantidad de prestaciones. Las **mismas serán asignadas a los diferentes recursos disponibles (móviles) en forma automática por parte de la aplicación**. Esta asignación la efectuará un algoritmo de optimización desarrollado a medida para esta problemática. Este algoritmo considerará las variables enunciadas a continuación para definir la asignación con la mejor calidad posible:

    - Localidad
    - Distancias entre móviles y prestaciones
    - Distancias entre las diferentes prestaciones
    - Carga de trabajo de móviles

- Afinidad tipo de móvil / prestación
- Relación médico / UCMQ (Relación de dependencia, tercero)
- Afinidad médico / prestación
- Tiempos de guardias
- Tipo de socio (socio directo, cápita, tercero, etc.)
- Uniformidad de uso de recursos (móviles)
- Cantidad mínima de prestación por móvil
- Reclamos de esa prestación
- Historial del cliente
- Cantidad de asignaciones que puede tener ese móvil
- Tiempo máximo de llegada estimado

Re-asignación

El proceso de re-asignación es de naturaleza manual. Cuando un despachador no acepte la asignación realizada por la aplicación, deberá asignarlo en forma manual. En el caso de prestaciones de código verde, el despachador podrá indicar que esa prestación sin asignación quede liberada para que la aplicación la asigne nuevamente o asignarla en forma manual. En el caso de prestaciones con código rojo o amarillo, si el prestador no acepta la asignación del sistema necesariamente la debe asignar en forma manual. En todos los casos que no se acepte la asignación del sistema, deberá indicarse un motivo. Ese motivo debe estar tabulado en la aplicación de Administración de Cabina.

Asignación a terceros

La asignación a terceros de códigos rojos y amarillos será realizada en forma manual. La aplicación informará cuando la localidad desde donde se genera el código rojo o amarillo, está fuera del área de cobertura.

La asignación a terceros de códigos verdes se efectuará en forma automática por parte de la aplicación, utilizando el algoritmo de optimización. Se considerarán las siguientes variables:

- Localidad
- Prestadores por localidad
- Afinidad prestador / prestación
- Tipo de socio (socio directo, cápita, tercero, etc.)
- Cantidad mínima de prestación por prestador
- Uniformidad de uso de recursos por prestador
- Reclamos de esa prestación

Seguimiento

El procedimiento estándar de trabajo implica que ante determinadas categorías de prestaciones (rojas y amarillas) y ante determinados tiempos de demora, se deberá realizar un seguimiento telefónico de las prestaciones desde la cabina, antes y después de la llegada del móvil al domicilio reportado. (Ver párrafo **Seguimiento Automatizado de Prestaciones**)

Pre-arribo

En las categorías de prestaciones que se indiquen en el sistema, y luego de determinado período de tiempo, el sistema alertará al despachador avisando que debe efectuar un llamado de pre-arribo. El despachador debe registrar en el sistema si realizó el mismo, detallado la información significativa. La aplicación identificará en la lista de prestaciones pendientes (con o sin asignación) cuáles requieren instrucciones de pre-arribo, diferenciando aquellas prestaciones que aún no tuvieron este tipo de llamados. (Ver **Seguimiento Automatizado de Prestaciones**)

Post-Arribo

En las categorías de prestaciones que se indiquen en el sistema, y luego de determinado período de tiempo, el sistema alertará al despachador avisando que debe efectuar un llamado de post-arribo. El despachador debe registrar en el sistema si realizó el mismo, detallado la información significativa. La aplicación identificará en la lista de prestaciones pendientes (con o sin asignación) cuales requieren instrucciones de post-arribo, diferenciando aquellas prestaciones que

aún no tuvieron este tipo de llamados. (Ver párrafo **Seguimiento Automatizado de Prestaciones**)

Derivación

Los códigos rojos y amarillos pueden implicar un traslado del paciente a un centro de internación. La definición de la realización del traslado surge del análisis que realizado por el médico en el lugar de la prestación

Lugares de derivación

Todos los centros de derivación estarán tabulados dentro de la aplicación. Asociados a cada centro se ingresará la coordenada donde se encuentra, además de la información de domicilio, teléfono y localidad. El móvil deberá informar cuando sale del domicilio, cuando llega al centro de derivación y cuando abandona el centro de internación. Esta información del estado del móvil deberá realizarse utilizando el sistema WAP-Nextel.

Seguimiento de la derivación

Luego de determinado período de tiempo y ante determinado evento, el sistema alertará al despachador avisando que debe efectuar un llamado de seguimiento de derivación. El despachador debe registrar en el sistema si realizó el mismo, detallado la información significativa. La aplicación identificará en la lista de derivaciones pendientes, diferenciando aquellas prestaciones que aún no tuvieron este tipo de llamados. (Ver párrafo **Seguimiento Automatizado de Prestaciones**)

Cierre

El cierre de la prestación se indicará utilizando el sistema Nextel-WAP, al igual que en la actualidad. El sistema alertará solicitando un llamado de seguimiento cuando las prestaciones en curso superen determinado tiempo, de acuerdo a la categoría de la prestación. El despachador debe registrar la realización de esa llamada, indicando el motivo de la tardanza. (Ver párrafo **Seguimiento Automatizado de Prestaciones**)

La figura 10.2 muestra la relación entre las diferentes etapas del proceso de atención de prestaciones y la relación con los tres módulos propuestos en este trabajo. El módulo de Cartografía Digital (GIS)

abarca el proceso completo, dado que se utiliza tanto para ubicar móviles y prestaciones como para ver información del estado y seguimiento de las prestaciones. El módulo de Asignación Automática se focaliza en la etapa de Despacho, y es el responsable de definir una asignación de móviles a prestaciones optimizada. El módulo de Seguimiento automatizado opera sobre las dos últimas etapas del proceso.

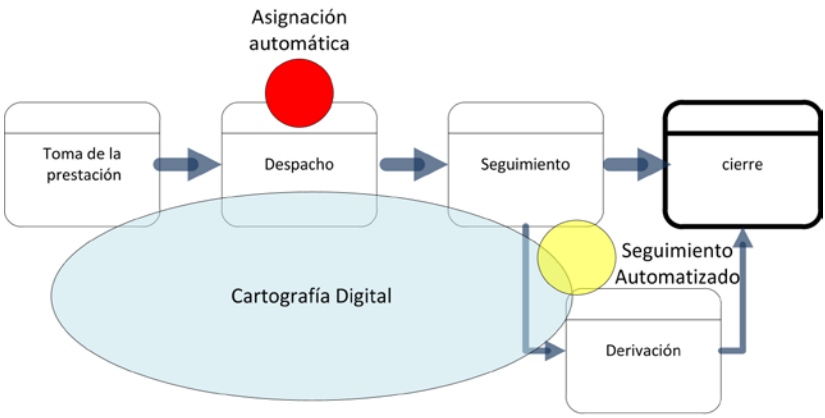


Figura 10.2: Proceso general de negocio y módulos propuestos

10.5 Cartografía Digital (GIS)

Se considera agregar el soporte de cartografía digital (GIS - *Geographic Information System*) al sistema de Administración de Cabina. El uso de cartografía tendrá dos objetivos principales:

1. Obtener las coordenadas exactas de la prestación (longitud y latitud) y demás ubicaciones (bases, centros de derivación, etc.), pudiendo además para el caso de las prestaciones que se toman en cabina validar la dirección en tiempo real.

2. Mostrar información del estado de prestaciones puntuales y cuadros de información general utilizando el soporte de mapas. De esta forma, se agrega al usuario la posibilidad de validar el estado de situación de una forma mucho más eficiente.

La figura 10.3 muestra un ejemplo del uso de mapas en el momento de la toma de la llamada, donde el operador puede ver la ubicación del domicilio y validar entre que calles se encuentra la propiedad.

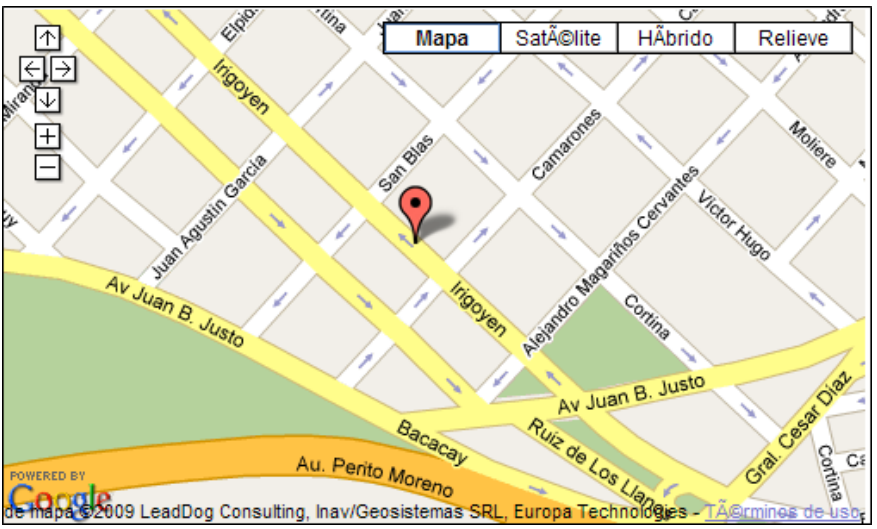


Figura 10.3: Geo-localización de prestaciones

La figura 10.4 muestra un ejemplo del uso de mapas en el momento de la asignación de un código rojo, donde el operador puede ver la ubicación del domicilio, los móviles más cercanos, y el tipo de móvil (UTIM en este caso) y el estado del mismo.

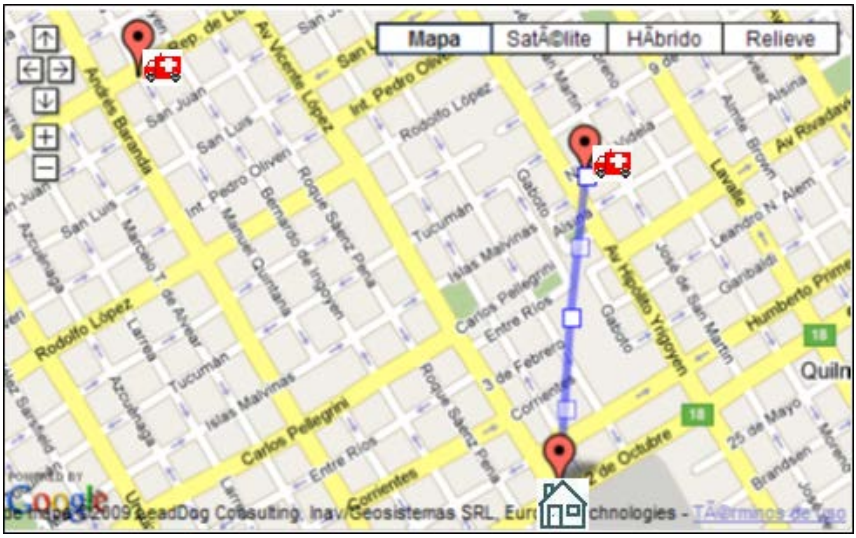


Figura 10.4: Calculo de distancia lineal entre móvil y prestaciones

A continuación se enumeran todas las instancias del proceso que tendrán apoyo cartográfico.

1. Toma de la Prestación

   En este punto del proceso el apoyo cartográfico tendrá como propósito dar soporte al operador de cabina para corroborar la información de domicilio, altura, localidad, intersecciones del lugar desde donde se genera la prestación. Adicionalmente, como punto final de este paso deben obtenerse las coordenadas exactas (longitud y latitud) del domicilio donde se reportó la prestación.

2. Asignación

   Se utilizará apoyo cartográfico en el momento de asignar móviles a prestaciones categorizadas como rojo o amarillo. La aplicación mostrará sobre un mapa digital la ubicación del domicilio donde se generó la prestación, el móvil asignado a la misma, y los tres o cuatro móviles más cercanos. Se utilizarán diferentes colores e indicadores para reflejar estado del móvil (en camino, en una prestación, etc.), tipo de móvil, categoría de la prestación y la exactitud de la ubicación.

3. Seguimiento

   El apoyo cartográfico en todas las tareas de seguimiento sin duda cobra un papel preponderante. Se propone la siguiente lista de consultas sobre mapas en base a lo relevado con los responsables del área.

   - Estado de una prestación categoría rojo o amarilla: Esta consulta nos mostrará en un mapa digital la ubicación del domicilio donde se generó la prestación, el móvil asignado a la misma, y los tres o cuatro móviles más cercanos. Se utilizarán diferentes colores e indicadores para reflejar estado del móvil (en camino, en una prestación, etc.), tipo de móvil, categoría de la prestación y la exactitud de la ubicación.

   - Móvil y asignaciones: Esta consulta nos mostrará el estado de un móvil en particular, donde se encuentra, el estado del mismo, y las asignaciones futuras que tiene asociadas.

   - Estado general de situación: Esta consulta nos mostrará sobre un mapa, y para cada localidad, la cantidad de prestaciones en curso y pendientes. Las prestaciones con categoría rojo y amarillo se mostrarán en forma individual. Las verdes se mostrarán agrupadas.

   - Excepciones: Esta consulta nos mostrará, sobre un mapa, las excepciones definidas por la Dirección Médica identificando las mismas con iconos y diferentes colores. El usuario podrá elegir que excepciones visualizar.

   - Prestaciones categorizadas como rojas con demora mayor a 15 minutos.

   - Prestaciones categorizadas como amarillas con demora mayor a 30 minutos.

   - Prestaciones categorizadas como verdes con demora mayor a 180 minutos.

   - Móvil ocupado en la misma prestación por más de 2 horas.

   - Móvil libre sin asignación por más de 2 horas.

- Móvil cuyo tiempo de comienzo de siguiente estado superó el doble de los estimado.
- Prestaciones a terceros: Esta consulta nos mostrará, sobre un mapa, las prestaciones asignadas a terceros mostrando tiempos de demora. Las prestaciones con categoría rojo y amarillo se mostrarán en forma individual. Las verdes se mostrarán agrupadas.

4. Derivación

   Para esta etapa se prevé un reporte que mostrará un mapa con todos los móviles que están realizando una derivación. La aplicación mostrará la ubicación del móvil, y el tiempo estimado hasta la concreción del siguiente estado.

5. Cierre

   Para esta etapa se prevé un reporte constituido por la cantidad de prestaciones cerradas por localidad. Esta consulta visualizará, sobre un mapa, la cantidad de prestaciones que se cerraron por cada localidad, en un rango de fechas determinado. Las prestaciones se agruparán por categoría. Se calculará para cada categoría y localidad la cantidad, tiempo promedio de resolución y varianza.

10.6 Asignación Automática

A los efectos de lograr un despacho automatizado, es condición necesaria ubicar todos los elementos en un plano de dos ejes cartesianos, longitud y latitud. Por lo tanto, antes de describir el proceso de asignación automática, se detalla cómo se ubican cada uno de los elementos que participan del proceso de despacho.

Cualquier elemento con los que se trabaja en la cabina se clasifica bien como un móvil (recurso), como una prestación (domicilio donde ocurre la prestación), como una base (ubicación fija de un móvil), o como alguna otra locación (centro de derivación). En esta sección de la tesis se hace mención a todo lo relacionado con la obtención y mantenimiento de las coordenadas (longitud y latitud) tanto de los domicilios de las prestaciones y locaciones en general así como de cada uno de lo móviles. En todo momento, todos los elementos (móviles con los que se cuenta y todas las locaciones) tienen asociadas una ubicación en el mapa. Como punto inicial es importante destacar

que todas las ubicaciones son aproximadas, incluso las ubicaciones indicadas por los dispositivos GPS, dado que las mismas tienen un margen de error y tienen un tiempo de demora. No es igual de exacta una ubicación calculada a partir de la información brindada por el GPS en el minuto actual, que una ubicación informada por el GPS cinco minutos atrás. Por lo tanto, cada ubicación tiene asociado un nivel de exactitud, siendo 1 el nivel más exacto y 10 (para el caso de ubicación de las prestaciones) o 20 (para el caso de ubicación de los móviles) el menos exacto. Debe quedar claro que todas las prestaciones así como todos los móviles siempre tienen una ubicación calculada, lo que varía es el grado de exactitud.

Ubicación de Prestaciones (Domicilios)

Los domicilios desde donde se generan los requerimientos de las prestaciones tienen asociadas un par de coordenadas que definirán su ubicación en el plano.

A las prestaciones recibidas las podemos clasificar en algunos de los siguientes tipos:

1. Llamados a cabina desde el lugar de la prestación: Con la toma del llamado, y apoyados en una herramienta de cartografía digital, se validará el domicilio reportado en línea y como paso final de esta tarea se obtendrán la longitud y latitud asociada. Esta forma de trabajo es válida para aquellas prestaciones que ingresan por una llamada telefónica directamente desde el lugar donde se genera la prestación.

2. Llamados a cabina desde otra cabina: Se seguirá el mismo procedimiento de trabajo intentando validar el domicilio y el resto de la información en línea.

3. Prestaciones que ingresan en forma electrónica (Internet): Existirá un operador que deberá como primera tarea, obtener las coordenadas del domicilio de la prestación con la información disponible.

Cualquier sea el caso anterior, si no es posible obtener la longitud y latitud del domicilio, se asociará con el mismo la coordenada asociada a la localidad (ver párrafo posterior **Mapa Discretizado**). La misma corresponde con el baricentro de la misma.

A continuación se detalla los diferentes niveles de exactitud que puede tener un cálculo de ubicación asociada a una prestación. La aplicación

intenta, en todos los casos, obtener y utilizar la ubicación con mayor nivel de exactitud (ver tabla 10.1).

Nivel de Exactitud	Concepto	Observaciones
2	Domicilio validado en línea en comunicación con el lugar de la prestación	Por el tomador de la cabina, utilizando la herramienta de cartografía digital y validando información en forma telefónica
3	Domicilio validado en línea en comunicación con otra cabina	Por el tomador de la cabina, utilizando la herramienta de cartografía digital y validando información con otra cabina tomadora, sin hablar directamente con el lugar de la prestación
5	Ubicación calculada con apoyo cartográfico, sin validación de domicilio	Por el tomador de la cabina, utilizando la herramienta de cartografía digital y buscando la ubicación por información brindada en forma directa desde la prestación
7	Ubicación calculada con apoyo cartográfico, sin validación de domicilio	Por el tomador de la cabina, utilizando la herramienta de cartografía digital y buscando la ubicación por información brindada a través de otra cabina
9	Baricentro de la localidad donde se encuentra el domicilio	Coordenadas asociadas al centro de la localidad (Buenos Aires) o Barrio (Capital Federal) asociada al domicilio

Tabla 10.1: Tabla de nivel de exactitud en la geo-localización de domicilios

Ubicación de los Móviles

Los móviles activos en cada momento siempre tienen una ubicación asociada. Estas coordenadas se calculan de diferentes formas, de acuerdo a la información disponible en cada momento. La ubicación del móvil tendrá asociado un nivel de exactitud, siendo 1 la más exacta, y 20 la menos exacta.

La aplicación intenta, en todos los casos, obtener y utilizar la ubicación mayor nivel de exactitud (ver tabla 10.2).

Nivel de exactitud	Concepto	Observaciones
1	Ubicación reportada por GPS, menor a 60 segundos	Coordenadas reportadas por el GPS
2	Ubicación calculada por código WAP	Coordenadas calculadas por código Nextel-WAP, a partir de lugar geo-localizado con exactitud 2
3	Ubicación calculada por código WAP	Coordenadas calculadas por código Nextel-WAP, a partir de lugar geo-localizado con exactitud 3
5	Ubicación calculada por código WAP	Coordenadas calculadas por código Nextel-WAP, a partir de lugar geo-localizado con exactitud 5
7	Ubicación calculada por código WAP	Coordenadas calculadas por código Nextel-WAP, a partir de lugar geo-localizado con exactitud 7
9	Ubicación calculada por código WAP	Coordenadas calculadas por código Nextel-WAP, a partir de lugar geo-localizado con exactitud 9
10	Ubicación reportada por GPS, mayor a 60 segundos y menor a 300 segundos	Coordenadas reportadas por el GPS
12	Ubicación calculada por código WAP, cálculo de distancia y tiempo	Coordenadas calculadas por código Nextel-WAP, cuando se informa la salida de un móvil a un lugar geo-localizado y se conoce el lugar y el mismo está geo localizado con nivel 2.

13	Ubicación calculada por código WAP, cálculo de distancia y tiempo	Coordenadas calculadas por código Nextel-WAP, cuando se informa la salida de un móvil a un lugar geo-localizado y se conoce el lugar y el mismo está geo localizado con nivel 3.
15	Ubicación calculada por código WAP, cálculo de distancia y tiempo	Coordenadas calculadas por código Nextel-WAP, cuando se informa la salida de un móvil a un lugar geo-localizado y se conoce el lugar y el mismo está geo localizado con nivel 5.
17	Ubicación calculada por código WAP, cálculo de distancia y tiempo	Coordenadas calculadas por código Nextel-WAP, cuando se informa la salida de un móvil a un lugar geo-localizado y se conoce el lugar y el mismo está geo localizado con nivel 7.
19	Ubicación calculada por código WAP, cálculo de distancia y tiempo	Coordenadas calculadas por código Nextel-WAP, cuando se informa la salida de un móvil a un lugar geo-localizado y se conoce el lugar y el mismo está geo localizado con nivel 9.
20	Baricentro de la localidad informada por el operador en forma manual	Coordenadas de la localidad informada por un operador

Tabla 10.2: Nivel de exactitud en la ubicación de móviles

Otras Locaciones

El resto de las locaciones que forman parte del proceso de despacho (bases, centros de derivación, etc.) están tabuladas dentro de la aplicación de Administración de Cabina, junto con sus coordenadas exactas calculadas por el proceso de geo-localización.

Mapa Discretizado

Con el propósito de ubicar todos los elementos que forman parte del proceso de despacho (prestaciones, móviles, bases, lugares de internación, etc.) sobre un plano, se hace necesario geo-localizar los mismos. El proceso de geo-localización consiste en calcular un par de

coordenadas (longitud - latitud) para cada uno de los elementos, en cada momento del tiempo.

Como se mencionó anteriormente en **Ubicaciones de Prestaciones y Móviles**, cada ubicación tiene un nivel de exactitud asociado. Un mapa discretizado se corresponde con una serie de ubicaciones (par ordenado latitud - longitud) representativas del área de cobertura. Estos puntos se corresponden con los baricentros de todas las localidades que conforman los partidos incluidos en el área de cobertura. Para el caso de Capital Federal, se consideraron los diferentes barrios que componen la comuna autónoma. Esta lista de ubicaciones estática se utilizará como elemento de última instancia para asociar una ubicación a alguno de los elementos que intervienen en el proceso de despacho. Por ejemplo, si sabemos que se decidió una derivación a un hospital ubicado en Villa Urquiza, Capital Federal, y no se cuenta con las coordenadas exactas de ese lugar de internación, se asociarán a ese destino de derivación las coordenadas correspondientes al baricentro de Villa Urquiza. Similar será el caso para aquellos domicilios correspondientes a prestaciones que por diferentes motivos (ingreso por Internet, imposibilidad de validación, etc.) no se pudieron geo-localizar.

Es importante destacar que el uso de estas ubicaciones pre-calculadas permiten realizar una proposición automática, mostrar información en un mapa y tomar decisiones, pero a costa de perder nivel de exactitud en el proceso.

Proceso de Asignación

Este módulo será el responsable de proponer los recursos (móvil o móviles) que atenderán cada prestación. Para realizar el cálculo, utilizará los siguientes elementos que forman parte del proceso de despacho.

- Ubicación de prestaciones y recursos (móviles)
- Tiempos desde la recepción de la prestación
- Categoría de la prestación (rojo, amarillo, verde)
- Síntoma
- Edad
- Estado del móvil (disponible, en camino, en domicilio, no disponible, etc.)

- Matriz de Afinidad Despachador / Zona Geográfica
- Nivel de Experiencia del Despachador
- Matriz de Afinidad Médico / Categoría de Prestación
- Matriz de Afinidad Médico / Edad del paciente
- Matriz de Afinidad Tipo de Móvil / Categoría de Prestación
- Utilización uniforme de recursos
- Tipo de contratación del médico
- Cantidad máxima de tiempo sin uso del recurso
- Cantidad mínima de prestaciones por día y por recurso
- Tiempos de guardias
- Tipo de socio (socio directo, cápita, tercero, etc.)
- Reclamos efectuados
- Prioridad cliente
- Historial de uso

Es claro ver que la aplicación utilizará la representación de estos elementos incorporadas dentro del sistema de Administración de Cabina, por lo que esta asignación será necesariamente aproximada, y actuará como una herramienta adicional para el despachador. En todo momento, el despachador podrá no aceptar la sugerencia propuesta por la aplicación, y realizar otra asignación conforme a su criterio.

El proceso a utilizar para realizar la proposición de asignación de móviles a prestaciones es un algoritmo del tipo de búsqueda y optimización, proveniente del campo de la inteligencia artificial. Estos tipos de algoritmos utilizan una función matemática de aptitud (fitness) que oficia como guía del proceso de búsqueda. Esta función de fitness es la que se debe optimizar. En este caso, se utilizará una técnica multi-objetivo, por lo que se optimizarán dos objetivos en forma simultánea, la reducción en los tiempos del servicio y la disminución de costos de operación.

Se comienza con una distribución al azar, y con el correr de las iteraciones se recorre el espacio de soluciones premiando aquellas propuestas con un fitness mayor. Entre cada una de las iteraciones, se utiliza un procedimiento para la construcción de nuevas soluciones

considerando las alternativas encontradas hasta el momento, incrementado la probabilidad de uso de aquellas soluciones encontradas con mejor aptitud, y agregando elementos aleatorios. Luego de una cantidad fija de iteraciones, y cuando no sea posible mejorar la solución alcanzada, el algoritmo se detendrá y propondrá el conjunto de soluciones no dominadas como solución al problema de optimización.

Cabe destacar que el proceso ofrecerá las soluciones en línea, lo que implica que la performance, confiabilidad y estabilidad del mismo debe ser muy elevada.

Otro factor a considerar es que el algoritmo siempre considera el estado de situación completo, tomando en cuenta para el cálculo todas las prestaciones y todos los recursos disponibles, y optimizando el proceso en su totalidad, no optimizando componentes individuales del sistema.

Configuración de Recursos

En este apartado se incluye todas las configuraciones e información adicional que administrará la aplicación de Administración de Cabina con el propósito de sugerir una asignación óptima de móviles a prestaciones.

Despachadores

Cada despachador tiene asignado un nivel de experiencia. La tabla 10.3 contiene los niveles de experiencia y las categorías de prestaciones que puede atender.

Experiencia	Categorías de Prestaciones
Alta	Rojo, Amarillo y Verde
Media	Amarillo y Verde
Baja	Verde

Tabla 10.3: Categorización experiencia despachadores

Cada despachador tiene asociado un área de influencia. La tabla 10.4 relaciona cada despachador con uno o más partidos.

Despachador	Partido	Afinidad (1 menor - 3 mayor)
YYYYYY	Avellaneda	1
	Quilmes	1
	Varela	3
XXXXXX	Lanús	3
	Avellaneda	3
	Lomas de Zamora	3

Tabla 10.4: Afinidad despachador - partido

Médicos

El sistema administra dos tablas relacionadas con los médicos responsables de cada móvil. La tabla 10.5 indica la afinidad entre cada médico y las categorías de las prestaciones y la tabla 10.6 la segunda entre cada médico y la edad de los pacientes.

Médico	Categoría Prestación	Afinidad (1 menor - 3 mayor)
YYYYYY	Rojo	1
	Amarillo	1
	Verde	3
XXXXXX	Rojo	3
	Amarillo	3
	Verde	3

Tabla 10.5: Matriz de afinidad médico - categoría de prestación

Médico	Edad	Afinidad (1 menor - 3 mayor)
YYYYYY	Niños (0 - 10)	3
	Jóvenes y Adultos (11 - 65)	2
	Ancianos (+ 65)	1
XXXXXX	Niños (0 - 5)	1
	Jóvenes y Adultos (11 - 65)	3
	Ancianos (+ 65)	3

Tabla 10.6: Matriz de afinidad médico - edad paciente

Para cada médico se indica qué tipo de relación lo une con la empresa (contratado o relación de dependencia). Se indica además si existe algún tipo de remuneración adicional por prestaciones, así como si tiene una cantidad mínima asegurada. Estos parámetros serán considerados al momento de la optimización de la asignación.

Síntomas

Existe un catálogo de la sintomatología existente. Asociado a cada síntoma, se definirá un nivel de severidad (1 poco severo / 10 muy severo). Esta severidad se utilizará para priorizar prestaciones clasificadas con la misma categoría (ver tabla 10.7).

Síntoma	Severidad
Dolor de Pecho	1
Fiebre	3
Mareos	5
Dolor de Cabeza	7

Tabla 10.7: Matriz de síntomas y severidad

Tipo de móviles

Cada móvil está asociado a un tipo de móvil. Para cada tipo de móvil de define una velocidad promedio de viaje y una varianza. Se administra una matriz en la aplicación que relacione los tipos de móviles y las diferentes categorías de prestaciones (ver tabla 10.8).

Tipo de Móvil	Categoría Prestación	Afinidad (1 menor – 3 mayor)
3*UTIM	Rojo	3
	Amarillos	3
	Verde	1
3*Auto Estándar	Rojo	1
	Amarillos	2
	Verde	3

Tabla 10.8: Matriz de tipo de móvil - categoría de la prestación

Lista de centros de derivación

Se administra en la aplicación una tabla con los diferentes centros de derivación, hospitales, sanatorios, y demás lugares de similares características. Para cada centro de derivación deberá constar el domicilio completo, teléfono, localidad, partido y provincia. El mismo deberá estar geo-localizado, a los efectos de disponer de información más exacta.

Lista de bases

Se administra en la aplicación una tabla con las diferentes bases de móviles. Para cada base deberá constar el domicilio completo, teléfono, localidad, partido y provincia. El mismo deberá estar geo-localizado, a los efectos de disponer de información más exacta.

Zona de Cobertura

Existe una tabla donde se definirá la zona de cobertura, y de que forma se atiende la demanda en la misma (ver tabla 10.9).

Partido	Cobertura
Quilmes	Mixta
Avellaneda	Propia
La Plata	Tercero (Detalle del tercero)

Tabla 10.9: Tabla de zona de cobertura

Motivos de rechazos de asignación

Existirá un catálogo de los motivos por los cuales se puede rechazar una asignación realizada por el sistema (ver tabla 10.10).

Motivo	Causa
Error de Asignación	Sistema
Asignación manual	Cabina
Error en ingreso de datos	Cabina

Tabla 10.10: Motivos de rechazo de una asignación

10.7 Seguimiento Automatizado de Prestaciones

Este módulo de la aplicación permite definir los avisos y/o alarmas dentro de la aplicación. De esta forma, podemos estandarizar los procedimientos de trabajo haciendo que el sistema guíe al despachador respecto de las acciones que debe realizar ante la ocurrencia de eventos pre-definidos.

Por ejemplo, se puede configurar en la aplicación de Administración de Cabina que se efectúe un llamado de pre-arribo para todas las prestaciones categorizadas como rojas, o que se realice un llamado manual al móvil cuando el tiempo en viaje supera la estimación del tiempo del traslado.

La tabla 10.11 contiene ejemplos de eventos, acciones y su configuración.

Categoría de Prestación	Estado	Evento	Tiempo	Acción	Respon-sable	Infor mar
Rojo	Asignada	Tiempo en ese estado	10 min.	Llamar al domicilio	Médico cabina	SI
Todas	En Camino / Derivación	Tiempo en exceso del estimado	10 min	Llamar a la unidad	Despa-chador	SI
Todas	En Domicilio	Tiempo en ese estado	30 min	Llamar a la unidad	Despa-chador	SI

Tabla 10.11: Eventos de seguimiento

Se tiene en consideración que esta herramienta sea sumamente amigable, y que no interfiera en demasía en las tareas habituales del despachador.

Es posible definir una fecha de vigencia desde - hasta para cada uno de las alarmas dentro de este módulo. De esta manera, es posible cambiar los procedimientos de trabajo de acuerdo a la disponibilidad de tiempo en cabina. Por ejemplo, en temporada alta se puede decidir no realizar algunos de los llamados de seguimiento, y se puede configurar la alarma en la aplicación pero que no sea obligatorio

registrar el resultado de la misma. Cada alarma tendrá un responsable de ejecución.

En este capítulo se presentó la solución ofrecida al problema en términos del negocio. Esta solución consiste en la optimización del proceso de despacho de móviles a prestaciones. Como primer etapa se hace necesario ubicar todos los elementos en juego en un plano de dos dimensiones, por lo que se hace necesario la incorporación de software para geo-localizar domicilios y móviles (GIS). Una vez que cada componente del problema tiene una ubicación asignada, podemos aplicar el algoritmo de optimización propiamente dicho. En el siguiente capítulo nos dedicamos en extenso a describir la solución algorítmica elegida, así como los experimentos realizados que nos permitieron llegar a esa elección. Por último, se propuso un módulo complementario para el seguimiento de la calidad de atención de la prestación.

Metaheurísticas Utilizadas

En este capítulo se describe específicamente la solución algorítmica que se utilizó para abordar este problema. Se detallan las metaheurísticas multi-objetivo seleccionadas, las modificaciones realizadas sobre las mismas, y los experimentos efectuados en tres instancias del problema del mundo real. En base a los resultados de la experimentación, se definió una solución híbrida que utiliza las tres metaheurísticas con mejor rendimiento.

11.1 Introducción

El núcleo del módulo de Asignación Automática es el algoritmo responsable de asignar que móvil atenderá a cada prestación. Como está detallado en los capítulos anteriores de la presente tesis, la asignación a las prestaciones de categorías rojas o amarillas se realizarán identificando los móviles más cercanos considerando su distancia lineal.
Para las prestaciones de categoría verde, se utilizará un módulo de asignación automática basado en metaheurísticas multi-objetivo. La naturaleza de este tipo de prestaciones es no programada, y representan más del 75% de las prestaciones totales. Este tipo de prestaciones no revisten peligro de vida. Desde el punto de vista de problema logístico y de estructura de costos las prestaciones pertenecientes a esta categoría representan el desafío mas importante para la compañía.
El objetivo central del módulo de asignación automática será la optimización de la utilización de los recursos, con el propósito de atender la mayor cantidad de prestaciones posibles, con un nivel de calidad adecuado, dado los recursos disponibles.

11.2 Modelado del problema

Los requerimientos iniciales de la compañía incluyen la optimización de dos objetivos contrapuestos, la reducción del tiempo de arribo a las prestaciones y minimización en el uso de equipos médicos de terceros (costos). Cada uno de estos objetivos tendrá una formula matemática asociada, y el algoritmo de optimización intentará minimizar ambas en forma simultánea. Aquellos individuos con valores menores en cada función objetivo serán mejores que aquellos con valores altos. Dado este contexto de proyecto, se definió la utilización de metaheurísticas multi-objetivo que permitan la optimización simultánea de ambos aspectos del problema. Este tipo de algoritmos retornan el conjunto de soluciones óptimas considerando todos los objetivos en juego. Este conjunto de soluciones es conocido como el frente de Pareto, y está formado por las soluciones no dominadas encontradas.
Se considera como parte importante de la solución propuesta que el tomador de decisiones pueda visualizar, una vez optimizado el problema, el conjunto de soluciones de alta calidad obtenido, y decidir cuál solución es la más adecuada. Asimismo, se prevé la posibilidad de que el sistema proponga una de las soluciones del frente de Pareto obtenido, de acuerdo a una configuración predefinida.
La visualización del frente de Pareto permite ganar un entendimiento mayor del problema al tomador de decisiones, así como la superficie del frente de Pareto revela la interacción entre los objetivos en oposición [230]. Es claro ver que se persigue en forma simultánea la mejora en la calidad de servicios y la optimización de los costos de operación de la empresa (figura 11.1).

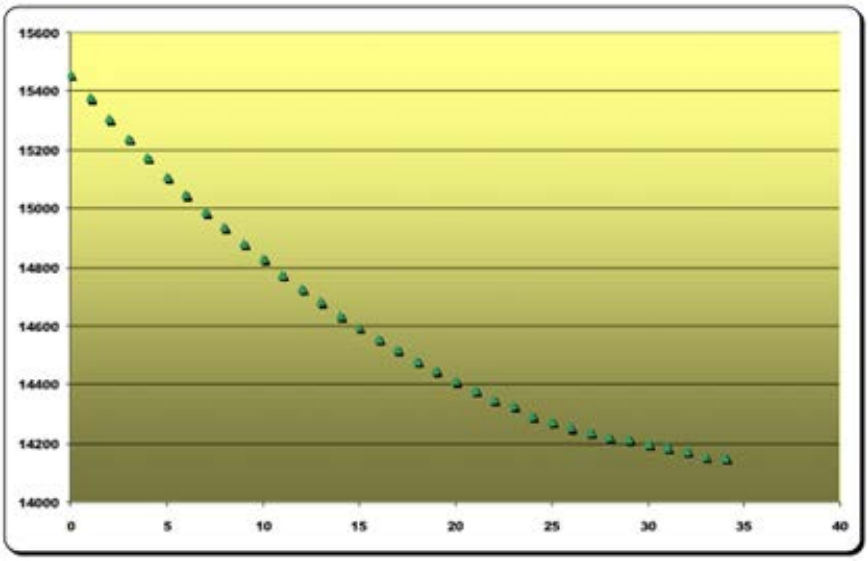


Figura 11.1: Visualización del frente de Pareto por parte del Tomador de Decisiones

El problema de optimización puede ser descrito identificando sus dos componentes principales, prestaciones médicas (pacientes que necesitan ser atendidos) y móviles médicos (equipos médicos y vehículos). En este capítulo se detallan los atributos principales de ambos componentes , a los efectos de clarificar el modelado del problema. En el capítulo de implementación se detallan la lista exhaustiva de atributos intervinientes.

Los atributos centrales de una prestación médica son:

Concepto	**Descripción**
Código (P1):	Identificación del servicio
Tiempo de llamada (P2):	Tiempo, en segundos, desde que se reportó el problema médico en la cabina de recepción de llamados
Ubicación (P3) :	Coordinadas, longitud y latitud, correspondiente a la ubicación geográfica donde se encuentra la prestación
Tiempo respuesta (P4):	Tiempo de respuesta promedio que se estima estar en el domicilio en función de la categoría de las prestación

Los móviles médicos son las unidades disponibles, de alta, media y baja complejidad, para atender las prestaciones médicas. Se caracterizan por los siguientes atributos principales:

Concepto	**Descripción**
Código (M1):	Identificación del móvil
Hora Desde (M2):	Tiempo, en segundos, a partir del cual el móvil va a estar disponible para su entrada en servicio. Se corresponde con el inicio de la hora de guardia de ese equipo médico
Hora Hasta (M3):	Tiempo, en segundos, a partir del cual el móvil dejará de estar disponible para su uso en servicio. Se corresponde con el fin de la hora de guardia de ese equipo médico
Ubicación (M4):	Coordinadas, longitud y latitud, de ubicación de ese vehículo. Esta ubicación es posible de calcularse con diferentes niveles de exactitud, según tabla en capítulo de implementación
Velocidad promedio (M5):	Velocidad promedio estimada de ese vehículo. La misma se corresponde a la velocidad lineal entre puntos de diferentes coordenadas
Dueño del vehículo (M6):	Atributo que identifica si el móvil pertenece a la compañía o es un servicio prestado por un proveedor (tercero)

Existen otros atributos que se detallan en el anexo de implementación.

El problema a resolver consiste en decidir cuál móvil médico se utilizará para atender cada una de las prestaciones, considerando el tiempo disponible de servicio de cada equipo médico (móvil), minimizando los tiempos de respuesta, y reduciendo el uso de equipos médicos de terceros.

Representación

Cada problema tiene su formulación matemática. Esta formulación puede ser visualizada graficando la función a optimizar para todos sus valores dentro del dominio de búsqueda. El “paisaje” que forma esta representación gráfica nos muestra el grado de complejidad de la función a optimizar. Los óptimos a encontrar se corresponden con los puntos máximo o mínimos de la función (figura 11.2).

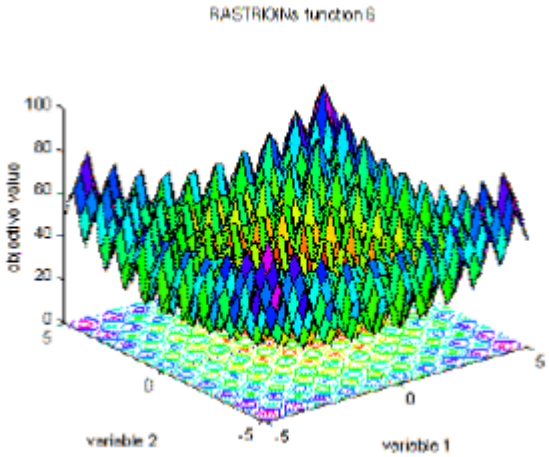


Figura 11.2: Ejemplo de función a optimizar con dos variables de decisión

La primera cuestión a resolver es la relacionada con la representación de las soluciones a utilizar para modelar el problema del mundo real. Esta representación debe considerar las características del problema en ciernes, y tiene una influencia central en la performance del algoritmo de optimización. Un mismo problema puede ser mas fácil o difícil de resolver dependiendo de la representación elegida [240].

Los algoritmos propuestos están concebidos, en su versión original, para trabajar en espacios de búsquedas continuos. Por lo tanto, en el ámbito de este trabajo se decidió optar por una representación continua (espacio de búsqueda) del espacio del problema. Esta decisión nos permite evaluar dentro del ámbito de este proyecto una mayor cantidad de metaheuristicas disponibles, y seleccionar aquellas con mejor desempeño evaluado. La utilización de otras representaciones, que normalmente traen aparejados el desarrollo de operadores específicos para las mismas no hubiera permitido el desarrollo del proyecto en el alcance especificado, tanto respecto de la variable tiempo como de la variable costo.

En esta tesis se propone utilizar la siguiente representación, que permite una representación matemática del problema abordado, y define un dominio de las funciones objetivos del tipo variables reales.

Cada individuo está representado por un vector de números reales. La dimensión del vector se corresponde con el número de prestaciones a atender. Por lo tanto, el tamaño del vector varia con la instancia del problema a resolver. El dominio de los valores para todas las dimensiones es el mismo y está relacionado con el número de móviles

médico disponibles. Formalmente, sea *m* el número de prestaciones y *n* el número de móviles médicos, cada individuo *I* es igual a un vector *m*-dimensional de números reales. El dominio de los valores de cada dimensión puede variar de 0,5 hasta n+0,4999.

Por ejemplo, un individuo podría tener los siguientes valores

$$A=(2.3546,0.6589,1.2357,1.7542)$$

La población completa estará constituida por *P* individuos de estas características, siendo *P* un parámetro inicial del algoritmo. Por ejemplo, una población formada por tres individuos podría ser de la siguiente forma:

$$A=(2.3546,0.6589,1.2357,1.754)$$

$$B=(1.9536,1.4500,0.8229,1.424)$$

$$C=(2.5546,3.8956,2.5700,3.7556)$$

Nótese que la representación elegida cumple como requisito el hecho de utilizar una representación real del espacio de búsqueda.

Decodificación del Vector Individuo

Cada dimensión del vector representa una prestación médica pendiente a atender. El valor real que toma cada dimensión representa el vehículo (móvil médico) responsable de atender cada prestación. El mismo se define aplicando la función redondeo() al valor real de cada dimensión. Esta función devuelve un valor entero que se corresponde con el sub-índice del móvil que atenderá esa prestación. La restricción en el dominio de los valores de cada dimensión asegura que la función siempre devuelve un sub-índice correspondiente a algún vehículo existente.

Cada vehículo puede atender mas de una prestación. El orden en que cada móvil atiende las diferentes prestaciones se define por el valor real, sin aplicar ningún tipo de función, y se considera de una manera ascendente. Si tomamos como ejemplo el individuo A de la tabla anterior, el equipo médico móvil número 1 atenderá las prestaciones 2 y 3, en ese orden, y el equipo médico 2 atenderá las prestaciones 4 y 1,

en ese orden. De forma análoga, tomando el individuo B, el móvil 1 atenderá las prestaciones 3, 4 y 2, en ese orden, y el móvil 2 atenderá las prestación 1. Nótese que siempre se asigna un móvil a una prestación. La inversa puede no ocurrir, pudiendo quedar móviles sin prestaciones asignadas.

Esta representación tiene la ventaja de una implementación simple y eficiente, dado que el uso de memoria requerido se incrementa linealmente con el número de prestaciones a atender, sin importar la cantidad de móviles disponibles. Dadas las características de hardware de los servidores actuales de rango medio, y en función del tamaño de la instancias de los problemas en el mundo real, el uso de memoria por parte de los algoritmos para representar la población de soluciones es despreciable.

La misma representación fue utilizada para todos los algoritmos evaluados.

Adicionalmente, cabe destacar que la relación establecida entre el espacio del problema y el espacio de búsqueda es del tipo inyectiva, dado que cada individuo representa una asignación única, pero cada asignación esta representada por mucho individuos.

Funciones Objetivo (Fitness Functions)

Las dos funciones objetivo a minimizar se definen como:

TE: Tiempo de espera promedio de las prestaciones
PT: Número de prestaciones que serán atendidas por móviles no propios

Con la primera de las funciones objetivo se intenta mejorar la calidad de servicio, dado que la variable tiempo de llegada para atender la prestación es la variable principal que define la calidad de atención, mas allá del acto médico en sí, que está fuera del alcance de este trabajo.

Con la segunda de las funciones se intenta aumentar la competitividad de la empresa optimizando la utilización de equipos médicos de terceros, dado que los mismos presentan una estructura de costos más importante medidos en términos comparativos con los equipos médicos propios, así como también se hace más difícil el aseguramiento en la calidad del servicio por ser personal médico y para-médico que no pertenece a la compañía.

Para estimar el tiempo de arribo a cada prestación, el vector que representa cada asignación, es en primera instancia organizado por móviles en orden creciente, y luego se ordena como se atenderán las prestaciones por parte de cada equipo médico, siguiendo lo establecido en el párrafo anterior.

El tiempo de espera se calcula de la siguiente manera para la primera prestación que atenderá el móvil:

$$TL=M2+TV1$$

$$TE=TL+P2$$

Siendo $TV1$ el tiempo de viaje, en segundos considerando $M4$, $M5$ y $P3$.

Para el resto de las prestaciones, el tiempo de espera se calcula de la siguiente forma:

$$TL=TEant+P4+TV2$$

$$TE=TL+P2$$

Siendo $TEant$ el TE de la prestación anterior y $TV2$ el tiempo de viaje desde la ubicación de la prestación anterior a la prestación actual, considerando $P3$, $M5$ y $P3ant$ ($P3$ de la prestación atendida previamente).

Luego, todos los tiempos de espera se suman y se promedian dividiendo los mismos por el número de prestaciones a atender por ese móvil. De esta forma, se continua con el resto de los móviles hasta decodificar el vector completo.

Para calcular la función objetivo 2 que se corresponde con la cantidad de prestaciones atendidas por móviles de terceros, simplemente se cuentan la cantidad de prestaciones que serán atendidas por móviles médicos no propios ($M6$).

Restricciones

En el planteamiento del problema hay restricciones a considerar en relación con la ventana de tiempo disponible (horario de guardia) de cada equipo médico. Cada móvil tiene un horario en el cual el mismo

no esta disponible, y no es posible asignarle prestaciones durante este rango de tiempo.

Para el manejo de restricciones, se creo una función de penalización. Esta penalización se suma al tiempo de espera (TE) calculado. De esta forma, si un móvil médico es asignado a una prestación fuera de su horario de disponibilidad, el tiempo de espera promedio (TE) se verá significativamente afectado.

La función de penalización se calcula como sigue:

$$Penalización = \begin{cases} 0 & M2 \leq TL \leq M3 \\ (M2 - TL)^2 & TL < M2 \\ (M3 - TL)^2 & TL > M3 \end{cases} \quad (11.1)$$

De esta forma, para aquellas asignaciones de móviles a prestaciones fuera de su horario de guardia activa la penalización es proporcional al cuadrado de la diferencia entre la hora asignada y la hora en que el móvil está disponible. Por lo tanto, cuando se asigna un móvil fuera de su horario de disponibilidad, pero cerca de sus extremos desde / hasta, la penalización será reducida. Sin embargo, cuando un móvil médico se asigna a una prestación fuera de su rango horario de disponibilidad, y lejos de los extremos horarios desde / hasta, la penalidad será mayor, desincentivando este escenario como posible solución.

A diferencia de funciones de penalización que dividen el espacio de búsqueda en soluciones factibles y no factibles, esta función de penalización permite obtener una aproximación de las soluciones en la frontera entre estos dos espacios, con acercamientos a estas regiones tanto desde el lado factible como desde el lado no factible [10].

11.3 Algoritmos

La experimentación se realizó utilizando 6 metaheurísticas representativas del estado del arte en optimización multi-objetivo, con la finalidad de medir los resultados comparados y seleccionar aquellas técnicas más apropiada para este problema.

Las metaheurisitcas seleccionadas fueron NSGAII[154], SPEA2 [155], PAES [218], varMOPSO [97], OMOPSO [225], y SMPSO [156]. Las primeras tres son técnicas referentes en este campo de estudio, las últimas tres son técnicas con buenos resultados reportados en problemas de laboratorio.

En el presente trabajo sólo se muestran los resultados obtenidos con NSGAII, SPEA2 y PAES dado que fueron aquellas técnicas que presentaron los mejores resultados en todas las instancias del problema, mostrando un desempeño superior que el resto de las técnicas evaluadas, en estos casos particulares. Es importante notar que todas las técnicas basadas en la metaheurística PSO mostraron un desempeño comparativo menor. En las conclusiones de la presente tesis se esboza una posible explicación de esta situación.

Operador de Mutación

Con el objetivo de mejorar la capacidad de búsqueda de las metaheurísticas utilizadas, se creo un operador de mutación para utilizar en todos los algoritmos evaluados. Este procedimiento es conceptualmente muy simple y fácil de implementar. Para el 1% de los casos, en forma previa a la evaluación de los individuos, se realizan cambios en los mismos. Estos cambios consisten en seleccionar el 5% de las dimensiones individuales de un individuo e intercambiar el valor de esa dimensión por el valor de otra seleccionada al azar. De esta forma, se intercambia el valor del 10% de las dimensiones para el 1% de los individuos. Los porcentajes precedentes fueron encontrados en forma empírica.

11.4 Experimentos Realizados

Para realizar la experimentación de las distintas metaheuristicas se realizaron optimizaciones de tres instancias del problema, correspondientes a situaciones reales provistas por la compañía.
La instancia denominada UCMQSmall se corresponde con un escenario compuesto por 37 prestaciones y 10 móviles de atención médica. La instancia UCMQMedium se corresponde a una instancia de 79 prestaciones y 18 móviles. La instancia UCMQLarge se corresponde a un escenario real de 233 prestaciones y 33 móviles médicos, entre equipos propios y de terceros.

Para los tres algoritmos NSGAII [154], SPEA2 [155] y PAES [218] se utilizó una población fija de 100 individuos. El tamaño del archivo externo de soluciones no dominadas fue de 100 individuos para todos los casos. En NSGAII [154] y SPEA2 [155] se utilizaron SBX y mutación polinomial como operadores de cruza y mutación, respectivamente. En PAES [218] también se utilizó mutación polinomial. La probabilidad de mutación fue 1/m, donde m es el número de las variables de decisión.

En todos los casos se realizó la evaluación de la metaheurística original y la misma metaheurística con el agregado del operador local de mutación implementado a nivel vector representativo del individuo (sufijo *pl* en los nombres de los algoritmos). A los efectos del experimento, se consideraron 100.000 evaluaciones de funciones, correspondientes al tiempo aproximado para resolver el problema en el mundo real. Se ejecutaron 30 corridas de prueba por cada algoritmo e instancia del problema.

Se calcularon los siguientes indicadores: *additive unary epsilon indicator*, *spread*, e *hipervolumen* [150] [231] [230], para comparar la calidad de los frentes de Pareto obtenidos.

Dado que son problemas del mundo real, los reales frentes de Pareto de los problemas son desconocidos. Los mismos fueron estimados por cada instancia del problema utilizando el algoritmo NSGAII y 1.000.000 de evaluación de funciones.

Para llevar a cabo la evaluación, se utilizó el framework de desarrollo jMetal [226], dada la importante cantidad de metaheurísticas implementadas, la buena documentación disponible y la generación automática de los indicadores de performance.

La tabla 11.1 muestra los resultados del indicador *Hipervolumen* (Mediana e IQR). Las tablas 11.2, 11.3 y 11.4 muestran si las diferencias son estadísticamente significantes. En cada tabla, el símbolo ▲ (peor) o símbolo △ (mejor) implica un $p-value<0.05$. indicando que la hipótesis nula (ambas distribuciones tiene la misma mediana) es rechazada; en otro caso, el símbolo - es usado.

Las figuras 11.3, 11.4 y 11.5 muestran los diagramas de caja para este indicador.

	UCMQLarge	UCMQMedium	UCMQSmall
NSGAII	$5{,}74e-02$	$6{,}27e-02$	$6{,}42e-01$
	$1{,}50e-01$	$1{,}60e-01$	$3{,}10e-02$
NSGAIIpl	$8{,}55e-02$	$1{,}00e-00$	$6{,}94e-01$
	$1{,}00e-01$	$0{,}00e-00$	$7{,}00e-02$
SPEA	$2{,}41e-01$	$2{,}81e-02$	$6{,}35e-01$
	$1{,}50e-01$	$3{,}10e-01$	$7{,}40e-02$
SPEApl	$2{,}37e-01$	$1{,}00e-00$	$7{,}15e-01$
	$1{,}20e-01$	$0{,}00e-00$	$6{,}20e-02$
PAES	$7{,}04e-01$	$2{,}55e-01$	$5{,}53e-01$
	$6{,}10e-02$	$5{,}50e-01$	$8{,}50e-02$
PAESpl	$0{,}00e+00$	$0{,}00e-00$	$1{,}08e-01$
	$0{,}00e+00$	$0{,}00e-00$	$1{,}00e-01$

Tabla 11.1: Hipervolumen. Mediana e IQR. Todas las Instancias

	NSGAIIpl	SPEA	SPEApl	PAES	PAESpl
NSGAII	–	▽	▽	▽	▲
NSGAIIpl		▽	▽	▽	▲
SPEA			–	▽	▲
SPEApl				▽	▲
PAES					▲
PAEpl					

Tabla 11.2: Hipervolumen. Significancia estadística. UCMQLarge

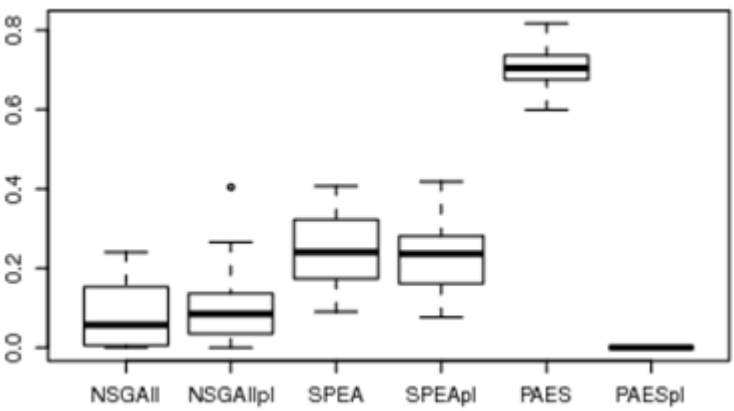


Figura 11.3: Hipervolumen. Diagrama de Cajas. UCMQLarge

	NSGAIIpl	SPEA	SPEApl	PAES	PAESpl
NSGAII	▽	–	▽	▽	▲
NSGAIIpl		▲	–	▲	▲
SPEA			▽	▽	▲
SPEApl				▲	▲
PAES					▲
PAEpl					

Tabla 11.3: Hipervolumen. Significancia estadística. UCMQMedium

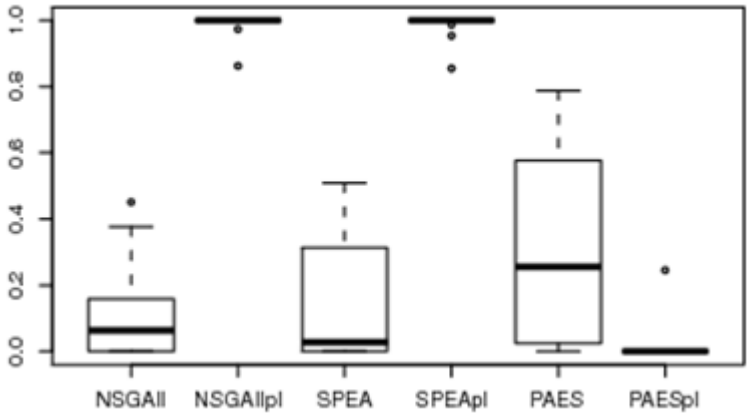


Figura 11.4: Hipervolumen. Diagrama de Cajas. UCMQMedium

	NSGAIIpl	SPEA	SPEApl	PAES	PAESpl
NSGAII	▽	–	▽	▲	▲
NSGAIIpl		▲	–	▲	▲
SPEA			▽	▲	▲
SPEApl				▲	▲
PAES					▲
PAEpl					

Tabla 11.4: Hipervolumen. Significancia estadística. UCMQSmall

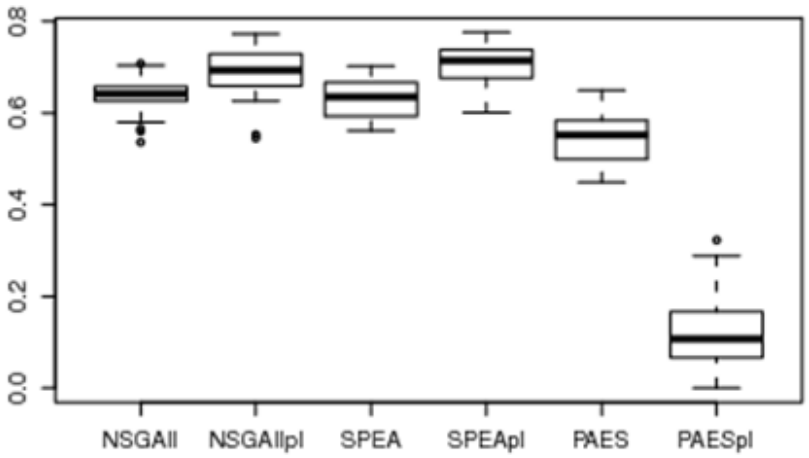


Figura 11.5: Hipervolumen. Diagrama de Cajas. UCMQSmall

La tabla 11.5 muestra los resultados del indicador *Epsilon* (Mediana e IQR). Las tablas 11.6, 11.7 y 11.8 muestran si las diferencias son estadísticamente significantes. Las figuras 11.6 a 11.8 muestran los diagramas de caja para este indicador.

	UCMQLarge	UCMQMedium	UCMQSmall
NSGAII	1,01e + 03	1,06e + 02	1,00e + 00
	4,60e + 02	1,20e + 02	5,50e + 00
NSGAIIpl	9,26e + 02	0,00e + 00	0,00e + 00
	3,30e + 02	0,00e + 00	1,00e + 00
SPEA	5,61e + 02	1,20e + 02	2,93e + 00
	2,50e + 02	1,30e + 02	6,00e + 01
SPEApl	6,02e + 02	0,00e + 00	0,00e + 00
	2,60E + 02	0,00e + 00	1,00e + 00
PAES	5,00e + 00	5,70e + 01	5,05e + 01
	3,00e + 00	1,30e + 02	8,90e + 01
PAESpl	3,52e + 05	9,84e + 02	8,40e + 02
	3,70e + 06	8,20e + 02	5,00e + 02

Tabla 11.5: Epsilon. Mediana e IQR. Todas las Instancias

	NSGAIIpl	SPEA	SPEApl	PAES	PAESpl
NSGAII	–	▽	▽	▽	▲
NSGAIIpl		▽	▽	▽	▲
SPEA			–	▽	▲
SPEApl				▽	▲
PAES					▲
PAEpl					

Tabla 11.6: Epsilon. Significancia estadística. UCMQLarge

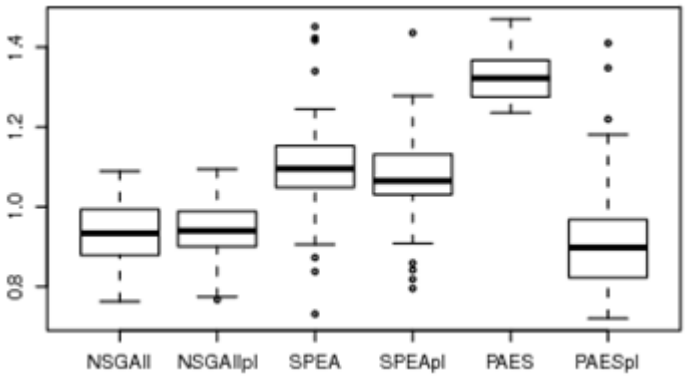


Figura 11.6: Epsilon boxplot. Large Instance

	NSGAIIpl	SPEA	SPEApl	PAES	PAESpl
NSGAII	▽	–	▽	▽	▲
NSGAIIpl		▲	–	▲	▲
SPEA			▽	▽	▲
SPEApl				▲	▲
PAES					▲
PAEpl					

Tabla 11.7: Epsilon. Significancia estadística. UCMQMedium

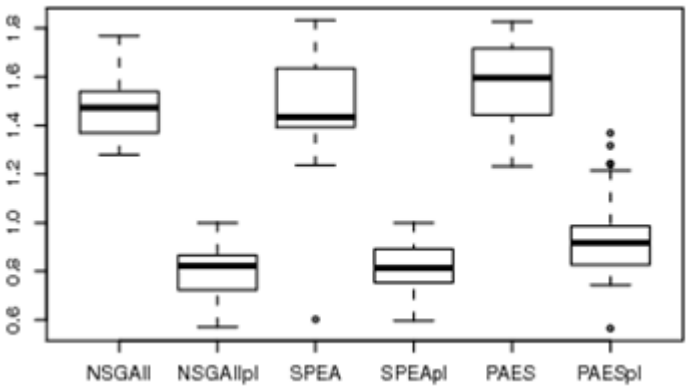


Figura 11.7: Epsilon boxplot. Medium Instance

	NSGAIIpl	SPEA	SPEApl	PAES	PAESpl
NSGAII	▽	–	▽	▲	▲
NSGAIIpl		▲	–	▲	▲
SPEA			▽	▲	▲
SPEApl				▲	▲
PAES					▲
PAEpl					

Tabla 11.8: Epsilon. Significancia estadística. UCMQSmall

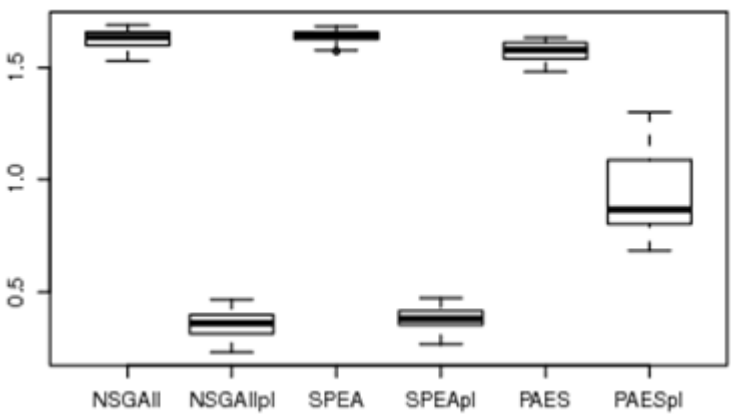


Figura 11.8: Epsilon. Diagrama de Cajas. UCMQSmall

La tabla 11.9 muestra los resultados del indicador *Spread* (Mediana e IQR). Las tablas 11.10, 11.11 y 11.12 muestran si las diferencias son estadísticamente significantes. Las figuras 11.9 a 11.11 muestran los diagramas de caja para este indicador.

	UCMQLarge	UCMQMedium	UCMQSmall
NSGAII	$9{,}33e-01$	$1{,}47e+00$	$1{,}63e+00$
	$8{,}5e-02$	$1{,}2e-01$	$3{,}8e-02$
NSGAIIpl	$9{,}34e-01$	$8{,}00e-01$	$3{,}58e-01$
	$8{,}5e-02$	$1{,}0e-01$	$6{,}2e-02$
SPEA	$1{,}10e+00$	$1{,}47e+00$	$1{,}64e+00$
	$1{,}6e-01$	$2{,}2e-01$	$2{,}6e-02$
SPEApl	$1{,}07e+00$	$8{,}15e-01$	$3{,}80e-01$
	$1{,}4e-01$	$9{,}6e-02$	$4{,}8e-02$
PAES	$1{,}33e+00$	$1{,}58e+00$	$1{,}57e+00$
	$6{,}9e-02$	$1{,}6e-01$	$4{,}2e-02$
PAESpl	$9{,}31e-01$	$9{,}52e-01$	$9{,}22e-01$
	$1{,}6e-01$	$1{,}8e-01$	$1{,}6e-01$

Tabla 11.9: Spread. Mediana e IQR. Todas las Instancias

Tabla 11.10: Spread. Significancia estadística. UCMQLarge

	NSGAII-pl	SPEA	SPEA-pl	PAES	PAES-pl
NSGAII	–	▲	▲	▲	–
NSGAIIpl		▲	▲	▲	–
SPEA			–	▲	▽
SPEApl				▲	▽
PAES					▽
PAESpl					

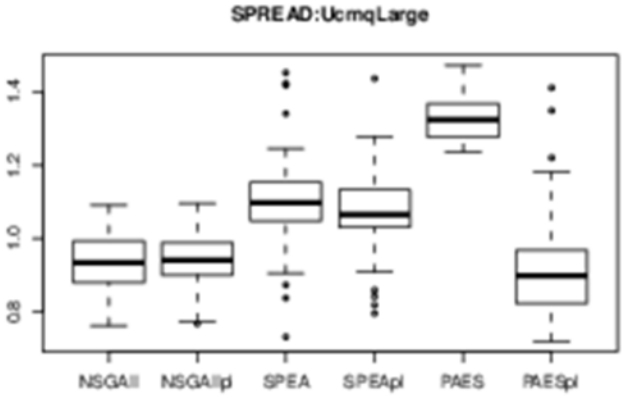


Figura 11.9: Spread boxplot. Large Instance

Tabla 11.11: Spread. Significancia estadísitica. UCMQMedium

	NSGAII-pl	SPEA	SPEA-pl	PAES	PAES-pl
NSGAII	▽	–	▽	▲	▽
NSGAIIpl		▲	–	▲	▲
SPEA			▽	▲	▽
SPEApl				▲	▲
PAES					▽
PAESpl					

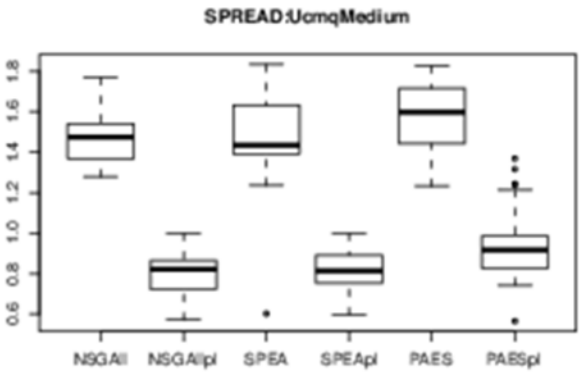


Figura 11.10: Spread boxplot. Medium Instance

Tabla 11.12: Spread. Significancia estadística. UCMQSmall

	NSGAII-pl	SPEA	SPEA-pl	PAES	PAES-pl
NSGAII	▽	–	▽	▽	▽
NSGAIIpl		▲	–	▲	▲
SPEA			▽	▽	▽
SPEApl				▲	▲
PAES					▽
PAESpl					

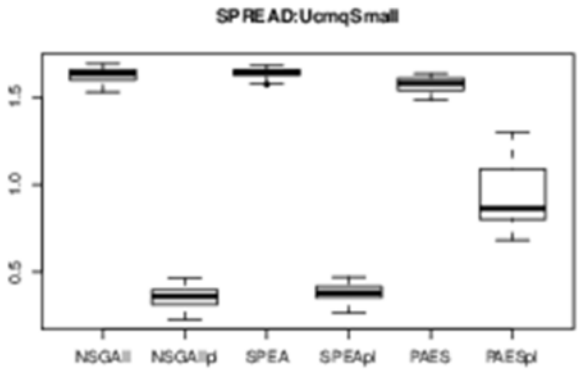


Figura 11.11: Spread boxplot. Small Instance

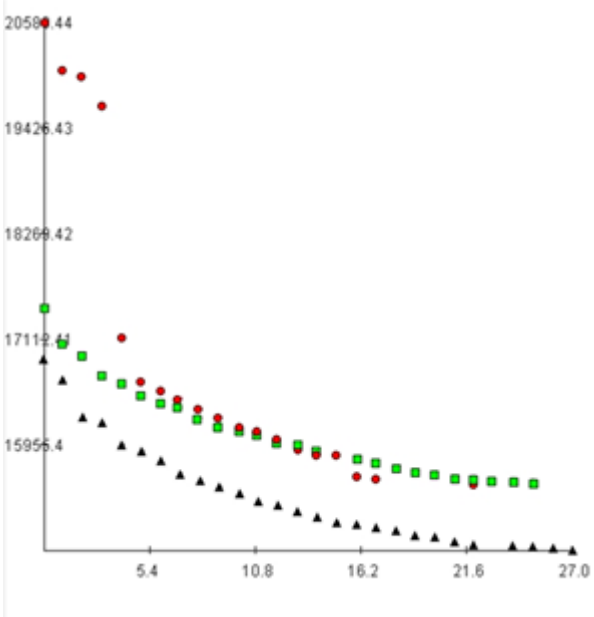


Figura 11.12: Frente de Pareto. UCMQLarge. Triángulos negros ⇒ PAES, Cuadrados verdes ⇒ SPEA2 , Círculos rojos ⇒ SPEA2 con mutación

11.5 Análisis de Resultados

Basados en los experimentos realizados, los siguientes aspectos pueden ser destacados:

Considerando el indicador *Hipervolumen*, para la instancia del problema mas grande (UCMQLarge), la metaheurística PAES produce los mejores resultados superando el rendimiento del resto de los algoritmos, con una diferencia estadísticamente significante. Para las instancias media (UCMQMedium) y pequeña (UCMQSmall) los mejores resultados son los obtenidos con los algoritmos NSGAII y SPEA2. Ambos con la utilización del operador de mutación propuesto.

Respecto del indicador *Epsilon*, para la instancia más grande, todas las metaheurísticas producen resultados similares, con la excepción de PAES con el procedimiento de mutación que mostró un bajo desempeño. Para las otras dos instancias, los mejores resultados son los obtenidos con los algoritmos NSGAII y SPEA2. Ambos con la utilización del operador de mutación propuesto.

Respecto del indicador *Spread*, para la instancia mayor del problema los mejores resultados fueron obtenidos por PAES con el agregado del

operador de mutación propuesto, seguido por NSGAII en las dos variantes evaluadas. Para las instancias mediana y pequeña del problema, los mejores resultados son los obtenidos con los algoritmos NSGAII y SPEA2. Ambos con la utilización del operador de mutación propuesto.

Analizando los indicadores que miden los rendimientos comparativos de las técnicas evaluadas podemos afirmar que para las instancias más grandes del problema el algoritmo PAES produce los mejores resultados. Asimismo, para las instancias medias y pequeñas la utilización de NSGAII y SPEA2 ofrecen los mejores desempeños. Además, puede apreciarse que en general el operador de mutación propuesto mejora los frentes de Pareto obtenidos por la versión original de las metaheurísticas.

En este caso particular se decidió utilizar las tres metaheurísticas evaluadas, cada una con la utilización y sin la utilización del operador de mutación propuesto, realizar una ejecución en paralelo con la instancia del problema real a resolver, y construir el frente de Pareto juntando los archivos de soluciones óptimas de cada algoritmo, manteniendo en el conjunto final sólo aquellas soluciones no dominadas. En la figura 11.12 se puede observar los frentes de Pareto obtenidos en una ejecución representativa de la instancia grande del problema y las tres metaheurísticas con mejor desempeño en este caso.

11.6 Conclusiones

En este capítulo se presentó una aplicación de metaheurísticas evolutivas en un caso del mundo real. El problema a resolver consistió en la automatización y optimización en tiempo real de las asignación de móviles médicos a prestaciones médicas. Se evaluaron un conjunto extenso de metaheurísticas evolutivas representativas del estado del arte en la materia. En base al problema a resolver, se seleccionaron aquellas técnicas que mostraron un mejor desempeño en las tres instancias del problema evaluado. Como solución al problema de optimización se compone un conjunto de soluciones no dominadas, formado por la unión de las soluciones encontradas por las tres metaheurísticas, en sus dos versiones. Estos procesos se corren en paralelo, en núcleos independientes de procesamiento.

El enfoque multi-objetivo de la solución propuesta también debe ser destacado, dado que es un enfoque innovador en este tipo de problemas. Ofreciendo un conjunto de soluciones óptimas (frente de Pareto), el TD (Tomador de decisiones) aprende acerca del problema y

puede visualizar cuantitativamente la relación entre los objetivos contrapuestos.

El uso de tecnología proveniente del campo de la investigación en aplicaciones de negocios es considerada sumamente importante. En este caso, existió una transferencia de conocimiento entre el mundo científico y el sector privado. El verdadero valor de la investigación reside en la aplicación del conocimiento científico para resolver problemas del mundo real.

Es importante destacar que este proyecto fue financiado por Unidad Coronaria Móvil Quilmes S.A., del grupo Ayuda Médica.

PARTE IV

Conclusiones

Conclusiones Finales

En este capítulo de la presente tesis se efectúa una revisión del trabajo realizado, destacando los puntos más relevantes de la presente investigación, profundizando en las lecciones aprendidas, y repasando aquellos aspectos pasibles de mejoras.
Se plantean además, las líneas de investigación a futuro que surgen a partir de los conocimientos y experiencias adquiridas durante el transcurso del presente trabajo.

12.1 Conclusiones Finales

Este trabajo se centra en el estudio y aplicación de metaheurísticas evolutivas aplicadas a problemas del mundo real. Estos algoritmos de optimización son, en su concepción matemática más elemental, algoritmos probabilísticos que generan muestras del espacio de búsqueda, intentando encontrar patrones de interés.

A partir del procesamiento de la información obtenida, se define la forma en que se direcciona la búsqueda. Esta elección está basada en el fuerte supuesto que las regularidades encontradas pueden servir de guía para encontrar los óptimos globales de la función. Dada la naturaleza de este tipo de algoritmos, creemos importante entender los elementos constitutivos de cualquier problema de búsqueda y optimización matemática. Estos elementos están siempre presentes y son comunes cualquiera sea la técnica que se aplique, de ahí su definición en los primeros capítulos del presente trabajo.

Luego de esta primera revisión de conceptos generales, la tesis se focaliza en las metaheurìsticas evolutivas. En especial, se profundiza el estudio de la metaheurística Optimización por Cúmulo de Partículas (PSO), técnica perteneciente a la rama conocida como Inteligencia de Enjambres, realizando una descripción y un estudio detallado de la misma.

Tomando como base la información obtenida se plantean diferentes versiones de algoritmos del tipo PSO, cada una con un fin muy específico, intentando en algunos casos suplir deficiencias encontradas y

en otros casos proponiendo variantes con mejoras a las versiones existentes.

En una primera instancia se plantea una versión paralela de la versión canónica del algoritmo, demostrando que naturalmente determinadas modalidades de paralelización son fácilmente implementables, y logrando interesantes índices de *speed-up*. Los resultados obtenidos en la comparación con la versión canónica muestran que la versión paralela mejora el rendimiento sólo en la minoría de las funciones de prueba utilizadas. Estos resultados son coincidentes con trabajos previos en la misma dirección, pero creemos que existe oportunidad de mejora de los resultados experimentando con diferentes configuraciones del algoritmo.

A continuación, en base al estudio del patrón de movimientos de las partículas cuando se encuentran cerca de un punto óptimo, se plantea una versión del algoritmo PSO que intenta reducir las oscilaciones de la partícula en su fase de aproximación final. La versión propuesta mejora el desempeño del algoritmo en la mayoría de las funciones de prueba utilizadas, mejora la velocidad de convergencia a los puntos óptimos, sin incrementar la tendencia a quedar atrapado en puntos sub-óptimos del espacio de búsqueda.

La siguiente versión desarrollada consiste en una propuesta de mejora sobre el algoritmo PSO aplicado a espacios binarios. Se trabajó en una nueva variante del cálculo de velocidad a partir del análisis de la versión binaria original del algoritmo y de otra variante propuesta por otros autores. Los objetivos pudieron ser alcanzados, y la versión propuesta obtuvo un rendimiento comparativo superior.

Hasta aquí, todas las variantes presentadas se aplicaron a optimización mono-objetivo.

La última versión de la técnica PSO presentada es una propuesta que incorpora el concepto de población de tamaño variable, y se aplicó a problemas multi-objetivo. Se evaluó esta alternativa comparándola con algoritmos de optimización representativos del estado del arte en optimización con más de un objetivo lográndose resultados destacados y superiores al resto de la técnicas evaluadas.

Respecto de estas actividades de laboratorio realizadas en el ámbito de la presente tesis podemos destacar los siguientes puntos:

1. Existen alternativas de mejoras a las diferentes técnicas existentes. Para poder proponer una versión con rendimiento superior, se hace necesario estudiar y entender la lógica propia de la técnica bajo estudio, y proponer mejoras orientadas a suplir falencias o mejorar puntos específicos de cada algoritmo.

2. Las mejoras propuestas implican, en algunos de los casos, aumentar la complejidad computacional de la técnica original. En estos casos se plantea un interesante *trade-off* entre resultados obtenidos y esfuerzo insumido cuya determinación de conveniencia dependerá de cada caso en particular.

3. La metaheurística Optimización por Cúmulo de Partículas (PSO) se muestra como una técnica flexible, que se puede aplicar tanto a espacios de búsqueda continuos como binarios, problemas mono-objetivo y multi-objetivo, versiones secuenciales y paralelas, en todos los casos con buenos resultados de desempeño, y sin grandes variaciones en sus componentes originales.

4. La adaptación del comportamiento de la metaheurística en función del problema a optimizar produce resultados positivos, dado que muchos de los parámetros de configuración de la misma son dependientes del problemas.

La última parte de este trabajo de investigación trata sobre la aplicación de metaheurísticas evolutivas a un problema del mundo real. En este caso, el problema a resolver fue la optimización de la asignación de móviles a prestaciones médicas, por parte de la principal empresa de emergencias del país. El caso de estudio se mostró sumamente interesante, dada la complejidad del problema en cuestión, así como la sensibilidad de la actividad de negocio de la empresa. Bajo el amparo de un contrato de transferencia entre la universidad y la empresa privada, fue posible desarrollar esta actividad alcanzando los objetivos propuestos. El proceso transcurrió de acuerdo a lo previsto, y los resultados del mismo para la compañía fueron muy beneficiosos.

Creemos importante destacar los siguientes aspectos del trabajo realizado:

1. Es de central importancia dedicar tiempo al entendimiento general del proyecto por parte de los involucrados. Este tipo de aplicaciones no son de uso habitual, aún en el ámbito de empresas con uso amplio de tecnología, y la comprensión del proyecto por parte de los niveles de dirección es parte fundamental del proceso.

2. Es fundamental encarar este tipo de proyectos en forma integral. Existen muchos problemas de optimización a resolver en las compañías, este es un caso acabado de los mismos, pero es importante entender que normalmente no se cuenta de antemano con toda la información que requieren los algoritmos de optimización para ser aplicados. Por lo tanto, es una parte principal del proyecto recabar la información faltante, e

instrumentar los mecanismos para que la misma sea recogida en forma precisa y regular como parte de los procesos de negocios estándares de la empresa.

3. Los problemas del mundo real son más complejos que los problemas de laboratorio. Se presentan dificultades relacionadas con resistencia al cambio, falta de entendimiento, carencia de comunicación, intereses creados, los que requieren ser gestionados para evitar el fracaso del proyecto.

4. Las instancias de los problemas del mundo real son desconocidas, y diferentes cada vez. Es una buena práctica experimentar con diferentes instancias reales del problema así como utilizar diferentes técnicas metaheurísticas, dado que a priori no es posible conocer cuál técnica se ajusta mejor al problema a optimizar. A la fecha, la base más firme para verificar cual técnica de optimización estocástica tiene mejor desempeño en un problema dado es el trabajo empírico.

5. Se debe aprovechar la capacidad de procesamiento disponible, paralelizando los procesos de optimización, utilizando las metaheurísticas que mejor desempeño mostraron en la etapa de experimentación. En este caso de estudio se detectaron tres metaheurísticas con dos variantes cada una con buenos resultados en las diferentes instancias. Por lo tanto se decidió ofrecer como solución al problema el frente de Pareto compuesto por las 6 variantes, lo que maximiza la robustez de la solución implementada.

6. Ofrecer al tomador de decisión el frente de Pareto obtenido es sumamente beneficioso, dado que permite que el mismo adquiera el conocimiento del problema y entienda el *trade-off* entre los objetivos en conflicto.

12.2 Trabajo Futuro

Luego del camino recorrido y del aprendizaje adquirido, surgen naturalmente líneas de investigación complementarias tanto en los ámbitos teóricos de experimentación en laboratorio como en casos del mundo real.

- Complejidad computacional: En las variantes de algoritmos del tipo PSO y MOPSO presentados en esta tesis se agregaron

procedimientos complementarios a las versiones originales que en la mayoría de los casos aumentaron sensiblemente la complejidad computacional del algoritmo. Sería adecuado investigar oportunidades de mejora en la implementación de los mismos tendientes a minimizar la complejidad de los algoritmos.

- Parámetros auto adaptables: Si bien esta línea de investigación es bastante amplía dentro de la rama de metaheurísticas evolutivas, creemos que es central para lograr algoritmos eficientes optimizando problemas complejos. Variar parámetros estándar de los algoritmos en función de la evolución de la búsqueda (coeficientes de aceleración, ponderación de las factores cognitivo y social) o aspectos más relevantes como el tamaño de la población o incluso el tipo de metaheurística es una línea de investigación con resultados prometedores. Este paradigma de pensamiento intenta utilizar soluciones robustas, probadas, pero que además sean capaces de maximizar la adaptación al problema en ciernes.

- Relación algoritmo / clase de problema: Esta línea de investigación se hace necesaria para entender porqué algoritmos con muy buenos resultados en algunos de los conjuntos de funciones de prueba presentan una pobre performance en una clase particular de problemas. En el ámbito de esta tesis se puede apreciar que todos los algoritmos del tipo MOPSO presentaron un bajo rendimiento en las instancias del problema del mundo real. Sin duda, la modelización de esta realidad configuró una clase particular de problemas donde los algoritmos multi-objetivo basados en PSO se ven superado por otras técnicas de búsqueda y optimización. En este caso en particular, si bien los experimentos realizados no fueron suficientes para aseverar esta afirmación, algunos indicios parecen indicar que los espacios de búsqueda con redundancia, como el seleccionado, disminuyen la capacidad de optimización de los algoritmos del tipo MOPSO, probablemente por su rápida convergencia a sectores específicos del espacio de búsqueda.

- Caso de estudio: Dado el éxito del proyecto desarrollado y la aceptación del mismo por parte de la compañía de emergencias, creemos que existen sendas oportunidades para seguir aplicando algoritmos de optimización en esta industria. Por ejemplo, la ubicación de los móviles de alta complejidad en función de la demanda en tiempo real (*Ambulance Location Problem*) es un problema típico para resolver con estas técnicas, así como la determinación de la cantidad de móviles y la ubicación de las bases de acuerdo a la demanda de servicios real e histórica.

PARTE V

Apéndice A

1. Rendimientos de algoritmos cPSO y dPSO

En esta sección se incluyen las tablas con los resultados del desempeño de los algoritmos cPSO y dPSO en las 25 funciones de prueba evaluadas según [208].

FES	Problema	01	02	03	04	05	06	07	08
1,00E+03	1	1,1042E+02	1,0685E+02	2,8346E+06	2,0838E+03	2,9015E+03	1,6182E+05	5,4743E+02	2,1238E+01
	7	1,7146E+02	4,2362E+02	6,6665E+06	6,2090E+03	4,4953E+03	5,6244E+05	8,4017E+02	2,1590E+01
	13	2,2970E+02	6,4973E+02	1,0929E+07	9,9651E+03	5,7154E+03	1,2492E+06	1,5663E+03	2,1687E+01
	19	3,8510E+02	8,6650E+02	1,8397E+07	1,5453E+04	7,4238E+03	2,4464E+06	4,1957E+03	2,1908E+01
	25	1,3452E+03	1,7054E+03	5,1505E+07	2,6027E+04	1,0122E+04	8,3756E+06	6,7320E+03	2,1978E+01
	Promedio	3,1200E+02	6,5485E+02	1,4077E+07	1,1002E+04	5,9758E+03	1,7582E+06	2,4867E+03	2,1698E+01
	Desvio	2,5988E+02	3,5901E+02	1,0712E+07	5,9380E+03	1,9371E+03	1,8298E+06	2,0190E+03	2,2958E-01
1,00E+04	1	4,2133E+01	9,6057E+01	5,3295E+05	2,5180E+02	9,4595E+02	1,6027E+05	1,1015E+00	2,0941E+01
	7	1,3164E+02	3,2763E+02	3,9865E+06	4,7025E+02	1,3154E+03	4,3170E+05	1,2576E+00	2,1652E+01
	13	1,5820E+02	4,8576E+02	5,2288E+06	6,3277E+02	1,5922E+03	5,1444E+05	1,3596E+00	2,1741E+01
	19	2,1207E+02	6,7665E+02	6,7361E+06	9,4525E+02	1,8501E+03	7,1691E+05	1,4950E+00	2,1896E+01
	25	3,1770E+02	1,4839E+03	1,6667E+07	2,1008E+03	3,0546E+03	3,1884E+06	1,8736E+00	2,2088E+01
	Promedio	1,6936E+02	5,3952E+02	5,7569E+06	7,6556E+02	1,6497E+03	8,1668E+05	1,3908E+00	2,1727E+01
	Desvio	6,6202E+01	3,3946E+02	3,3999E+06	4,5438E+02	4,5392E+02	8,0384E+05	1,8109E-01	2,4990E-01
1,00E+05	1	0,0000E+00	0,0000E+00	4,2351E+03	3,7053E-03	1,1130E-09	4,0208E-01	3,9382E-02	2,0022E+01
	7	0,0000E+00	1,0000E-12	3,1239E+04	9,3156E-03	4,8004E-06	3,2415E+00	6,4002E-02	2,0175E+01
	13	0,0000E+00	2,0000E-12	5,4490E+04	5,7834E-01	2,1601E-04	5,5833E+00	1,2067E-01	2,0302E+01
	19	0,0000E+00	7,0000E-12	1,1646E+05	1,4080E+00	2,5663E-03	1,6205E+01	1,4782E-01	2,0580E+01
	25	3,1200E-10	1,2968E-08	2,3408E+05	1,1756E+01	4,8889E-02	3,7152E+02	4,0111E-01	2,0669E+01
	Promedio	1,3360E-11	6,0396E-10	7,5450E+04	1,7889E+00	8,0043E-03	3,7364E+01	1,3862E-01	2,0351E+01
	Desvio	6,2275E-11	2,6040E-09	6,3720E+04	3,2989E+00	1,5552E-02	8,3575E+01	1,0305E-01	2,0429E-01

Tabla 1: Funciones de la 01 a la 08 cPSO en 10 dimensiones

FES	Problema	09	10	11	12	13	14	15	16
1,00E+03	1	1,2775E+02	1,0671E+02	1,7809E+01	2,5819E+05	4,6781E+04	4,5395E+00	9,2135E+02	6,2995E+02
	7	1,6370E+02	2,1453E+02	1,9696E+01	4,6492E+05	1,9744E+06	4,7147E+00	1,2083E+03	8,0266E+02
	13	1,8025E+02	2,9944E+02	1,9696E+01	5,9248E+05	6,8302E+06	4,9560E+00	1,4939E+03	1,0056E+03
	19	1,9513E+02	3,9073E+02	2,3814E+01	8,3034E+05	9,5793E+06	5,1992E+00	1,7944E+03	1,5368E+03
	25	2,6322E+02	5,4893E+02	2,4424E+01	1,6766E+06	1,6660E+07	5,3924E+00	2,4195E+03	2,1619E+03
	Promedio	1,8238E+02	3,0659E+02	2,1112E+01	6,7508E+05	6,1107E+06	4,9733E+00	1,5093E+03	1,1672E+03
	Desvio	3,2866E+01	1,1196E+02	2,2259E+00	3,3300E+05	5,0424E+06	2,7195E-01	3,7180E+02	4,6498E+02
1,00E+04	1	1,3026E+02	1,4305E+02	1,7138E+01	2,6936E+05	1,7653E+04	4,5262E+00	9,4139E+02	5,1436E+02
	7	1,8431E+02	2,1179E+02	1,8471E+01	5,6415E+05	1,2278E+06	4,8063E+00	1,2021E+03	8,0322E+02
	13	2,1056E+02	2,8634E+02	1,9696E+01	6,7741E+05	3,0697E+06	5,0291E+00	1,3155E+03	1,0108E+03
	19	2,3891E+02	3,5395E+02	2,2033E+01	8,5945E+05	5,4347E+06	5,1680E+00	1,6543E+03	1,3409E+03
	25	3,1838E+02	5,5791E+02	2,5144E+01	1,3597E+06	2,4196E+07	5,4774E+00	2,4363E+03	2,5145E+03
	Promedio	2,1116E+02	2,9122E+02	2,0312E+01	7,1186E+05	4,2883E+06	5,0116E+00	1,4676E+03	1,1232E+03
	Desvio	4,3805E+01	1,0314E+02	2,3336E+00	2,6507E+05	5,0915E+06	2,8821E-01	4,0885E+02	4,8987E+02
1,00E+05	1	8,9546E+00	1,8904E+01	4,3410E+00	9,5500E-01	1,5626E+00	3,0932E+00	4,0000E+02	1,2020E+02
	7	1,9899E+01	3,4824E+01	4,9359E+00	3,5461E+01	1,9406E+00	4,0929E+00	4,2510E+02	1,4646E+02
	13	3,2834E+01	4,6764E+01	6,3265E+00	2,2450E+03	2,7336E+00	4,2685E+00	4,3383E+02	1,7754E+02
	19	3,9798E+01	5,6712E+01	7,6326E+00	1,2457E+04	3,6944E+00	4,4807E+00	4,5481E+02	2,2637E+02
	25	1,0845E+02	9,7504E+01	8,6200E+00	2,9142E+04	8,2730E+00	4,5076E+00	5,8231E+02	4,2446E+02
	Promedio	3,5139E+01	4,8346E+01	6,4055E+00	6,1507E+03	3,0717E+00	4,2043E+00	4,5708E+02	1,9616E+02
	Desvio	2,1004E+01	2,0344E+01	1,3739E+00	7,8979E+03	1,5451E+00	3,4259E-01	5,4065E+01	7,1043E+01

Tabla 2: Funciones de la 09 a la 16 cPSO en 10 dimensiones

FES	Problema	17	18	19	20	21	22	23	24	25
1,00E+03	1	4,8450E+02	1,3230E+03	1,2706E+03	1,3001E+03	1,4088E+03	1,1050E+03	1,4557E+03	1,3716E+03	6,3106E+02
	7	9,8952E+02	1,4790E+03	1,5018E+03	1,6359E+03	1,6441E+03	1,6371E+03	1,6451E+03	1,6079E+03	1,3523E+03
	13	1,2118E+03	1,6329E+03	1,7301E+03	1,8848E+03	1,6954E+03	2,8126E+03	1,7094E+03	1,6424E+03	1,7401E+03
	19	1,6316E+03	1,9065E+03	1,9761E+03	1,9987E+03	1,8289E+03	4,3612E+03	1,8723E+03	1,7467E+03	2,0365E+03
	25	2,9897E+03	3,0542E+03	2,9686E+03	2,9393E+03	2,1713E+03	1,3459E+05	2,2277E+03	2,2255E+03	2,4022E+04
	Promedio	1,3664E+03	1,7543E+03	1,7815E+03	1,8723E+03	1,7294E+03	8,8693E+03	1,7729E+03	1,6742E+03	2,8687E+03
	Desvio	5,9451E+02	3,9973E+02	3,8474E+02	3,4346E+02	1,7294E+02	2,6376E+04	2,0088E+02	1,7706E+02	4,5270E+03
1,00E+04	1	5,7891E+02	1,2172E+03	1,2992E+03	1,1908E+03	1,3864E+03	1,1953E+03	1,4319E+03	1,3875E+03	Nan
	7	1,0241E+03	1,5953E+03	1,3575E+03	1,3748E+03	1,5727E+03	1,6327E+03	1,6170E+03	1,4989E+03	1,0045E+03
	13	1,3477E+03	1,7317E+03	1,6297E+03	1,5367E+03	1,6441E+03	2,3395E+03	1,7235E+03	1,5476E+03	1,7811E+03
	19	2,0176E+03	1,9773E+03	2,0044E+03	1,9118E+03	1,7884E+03	3,4547E+03	1,8645E+03	1,6810E+03	1,3245E+04
	25	2,9994E+03	2,4595E+03	2,8607E+03	4,0205E+03	2,3028E+03	2,9890E+04	2,1501E+03	1,9297E+03	Nan
	Promedio	1,5114E+03	1,7524E+03	1,7614E+03	1,7234E+03	1,6861E+03	3,9614E+03	1,7298E+03	1,5868E+03	2,1177E+03
	Desvio	6,3780E+02	3,0336E+02	4,6140E+02	5,9774E+02	2,0948E+02	5,8795E+03	1,7988E+02	1,3904E+02	2,8537E+03
1,00E+05	1	1,7509E+02	3,4754E+02	3,0000E+02	3,4736E+02	5,0000E+02	3,3275E+02	5,5947E+02	2,0000E+02	4,0451E+02
	7	2,8917E+02	5,0709E+02	4,3394E+02	5,7186E+02	5,0000E+02	7,9077E+02	6,9865E+02	2,0000E+02	4,1918E+02
	13	3,3472E+02	8,0009E+02	7,0248E+02	8,0010E+02	5,0000E+02	8,0304E+02	9,3137E+02	2,0000E+02	4,2282E+02
	19	3,9454E+02	9,1401E+02	8,0024E+02	9,7844E+02	9,0780E+02	8,2984E+02	1,1823E+03	2,0000E+02	Nan
	25	5,2984E+02	1,0350E+03	1,0377E+03	1,0544E+03	1,1838E+03	9,2888E+02	1,2892E+03	5,0000E+02	Nan
	Promedio	3,4441E+02	7,2664E+02	6,4628E+02	7,6938E+02	6,9867E+02	7,8711E+02	9,3410E+02	2,1200E+02	4,1878E+02
	Desvio	9,4060E+01	2,3048E+02	2,5931E+02	2,3889E+02	2,6281E+02	1,3559E+02	2,5480E+02	6,0000E+01	5,9096E+00

Tabla 3: Funciones de la 17 a la 25 cPSO en 10 dimensiones

Problema	1	7	13	19	25	Promedio	Desvio	Exito
01	5,3482E+04	5,2634E+04	5,2161E+04	5,1321E+04	5,0314E+04	5,1958E+04	9,0573E+02	100%
02	5,4177E+04	5,2276E+04	5,1683E+04	5,0825E+04	5,0144E+04	5,1669E+04	1,1123E+03	96%
03	-	-	-	-	-	-	-	0%
04	-	-	-	-	-	-	-	0%
05	-	-	-	-	-	-	-	0%
06	-	-	-	-	-	-	-	0%
07	-	-	-	-	-	-	-	0%
08	-	-	-	-	-	-	-	0%
09	-	-	-	-	-	-	-	0%
10	-	-	-	-	-	-	-	0%
11	-	-	-	-	-	-	-	0%
12	-	-	-	-	-	-	-	0%
13	-	-	-	-	-	-	-	0%
14	-	-	-	-	-	-	-	0%
15	-	-	-	-	-	-	-	0%
16	-	-	-	-	-	-	-	0%
17	-	-	-	-	-	-	-	0%
18	-	-	-	-	-	-	-	0%
19	-	-	-	-	-	-	-	0%
20	-	-	-	-	-	-	-	0%
21	-	-	-	-	-	-	-	0%
22	-	-	-	-	-	-	-	0%
23	-	-	-	-	-	-	-	0%
24	-	-	-	-	-	-	-	0%
25	-	-	-	-	-	-	-	0%

Tabla 4: Funciones de la 01 a la 25 cPSO en 10 dimensiones

FES	Problema	01	02	03	04	05	06	07	08
1,00E+03	1	1,1042E+02	1,0685E+02	2,8346E+06	2,0838E+03	2,9015E+03	1,6182E+05	5,4743E+02	2,1238E+01
	7	1,7146E+02	4,2362E+02	6,6665E+06	6,2090E+03	4,4953E+03	5,6244E+05	8,4017E+02	2,1590E+01
	13	2,2970E+02	6,4973E+02	1,0929E+07	9,9651E+03	5,7154E+03	1,2492E+06	1,5663E+03	2,1687E+01
	19	3,8510E+02	8,6650E+02	1,8397E+07	1,5453E+04	7,4238E+03	2,4464E+06	4,1957E+03	2,1908E+01
	25	1,3452E+03	1,7054E+03	5,1505E+07	2,6027E+04	1,0122E+04	8,3756E+06	6,7320E+03	2,1978E+01
	Promedio	3,1200E+02	6,5485E+02	1,4077E+07	1,1002E+04	5,9758E+03	1,7582E+06	2,4867E+03	2,1698E+01
	Desvio	2,5988E+02	3,5901E+02	1,0712E+07	5,9380E+03	1,9371E+03	1,8298E+06	2,0190E+03	2,2958E-01
1,00E+04	1	4,2133E+01	9,6057E+01	5,3295E+05	2,5180E+02	9,4595E+02	1,6027E+05	1,1015E+00	2,0941E+01
	7	1,3164E+02	3,2763E+02	3,9865E+06	4,7025E+02	1,3154E+03	4,3170E+05	1,2576E+00	2,1652E+01
	13	1,5820E+02	4,8576E+02	5,2288E+06	6,3277E+02	1,5922E+03	5,1444E+05	1,3598E+00	2,1741E+01
	19	2,1207E+02	6,7665E+02	6,7361E+06	9,4525E+02	1,8501E+03	7,1691E+05	1,4950E+00	2,1896E+01
	25	3,1770E+02	1,4839E+03	1,6667E+07	2,1008E+03	3,0546E+03	3,1884E+06	1,8736E+00	2,2088E+01
	Promedio	1,6936E+02	5,3952E+02	5,7569E+06	7,6556E+02	1,8497E+03	8,1688E+05	1,3908E+00	2,1727E+01
	Desvio	6,6202E+01	3,3946E+02	3,3999E+06	4,5438E+02	4,5392E+02	8,0384E+05	1,8109E-01	2,4990E-01
1,00E+05	1	0,0000E+00	0,0000E+00	4,2351E+03	3,7053E-03	1,1130E-09	4,0208E-01	3,9382E-02	2,0022E+01
	7	0,0000E+00	1,0000E-12	3,1239E+04	9,3156E-03	4,8004E-06	3,2415E+00	6,4002E-02	2,0175E+01
	13	0,0000E+00	2,0000E-12	5,4490E+04	5,7834E-01	2,1601E-04	5,5833E+00	1,2067E-01	2,0302E+01
	19	0,0000E+00	7,0000E-12	1,1646E+05	1,4080E+00	2,5863E-03	1,6205E+01	1,4782E-01	2,0580E+01
	25	3,1200E-10	1,2968E-08	2,3408E+05	1,1756E+01	4,8889E-02	3,7152E+02	4,0111E-01	2,0669E+01
	Promedio	1,3360E-11	6,0396E-10	7,5450E+04	1,7889E+00	8,0043E-03	3,7364E+01	1,3862E-01	2,0351E+01
	Desvio	6,2275E-11	2,6040E-09	6,3720E+04	3,2989E+00	1,5552E-02	8,3575E+01	1,0305E-01	2,0429E-01

Tabla 5: Funciones de la 01 a la 08 cPSO en 10 dimensiones

FES	Problema	09	10	11	12	13	14	15	16
1,00E+03	1	5,6041E+02	6,1852E+02	5,2156E+01	2,5340E+06	8,4297E+05	1,4235E+01	1,2756E+03	1,2684E+03
	7	6,4549E+02	1,0223E+03	5,7242E+01	3,6524E+06	7,5851E+06	1,4822E+01	1,4320E+03	1,4186E+03
	13	6,9998E+02	1,2368E+03	5,9904E+01	4,0703E+06	1,4375E+07	1,5097E+01	1,5278E+03	1,6994E+03
	19	8,0525E+02	1,3571E+03	5,9904E+01	5,1496E+06	2,0727E+07	1,5202E+01	1,7870E+03	1,8690E+03
	25	9,0116E+02	1,7224E+03	6,8739E+01	7,5250E+06	3,9058E+07	1,5581E+01	2,0182E+03	2,1970E+03
	Promedio	7,1726E+02	1,2329E+03	5,8634E+01	4,5732E+06	1,5490E+07	1,5000E+01	1,6087E+03	1,6685E+03
	Desvio	9,9976E+01	2,6681E+02	3,8378E+00	1,3731E+06	9,9284E+06	3,5063E-01	2,1743E+02	2,6442E+02
1,00E+04	1	5,4625E+02	7,7199E+02	5,3112E+01	2,9063E+06	2,3069E+06	1,4654E+01	1,3357E+03	1,0909E+03
	7	6,4216E+02	1,1238E+03	5,7539E+01	4,0244E+06	4,2326E+06	1,4782E+01	1,4784E+03	1,3151E+03
	13	7,2232E+02	1,3112E+03	5,9904E+01	4,5661E+06	8,3956E+06	1,5009E+01	1,6025E+03	1,5245E+03
	19	7,6557E+02	1,3831E+03	6,1186E+01	6,0098E+06	1,2327E+07	1,5293E+01	1,8195E+03	1,7509E+03
	25	8,7449E+02	1,4907E+03	6,9608E+01	7,0233E+06	2,7839E+07	1,5505E+01	2,4276E+03	1,9970E+03
	Promedio	7,0862E+02	1,2594E+03	5,9790E+01	4,8598E+06	9,9767E+06	1,5041E+01	1,6760E+03	1,5440E+03
	Desvio	8,3958E+01	1,8255E+02	4,2127E+00	1,3038E+06	7,2074E+06	2,7989E-01	2,9045E+02	2,7176E+02
1,00E+05	1	4,8324E+02	6,6272E+02	5,5222E+01	2,5820E+06	1,5126E+06	1,4352E+01	9,5689E+02	7,0371E+02
	7	5,5312E+02	8,6097E+02	5,8879E+01	3,5093E+06	2,9452E+06	1,4784E+01	1,1923E+03	8,8598E+02
	13	5,9366E+02	9,5141E+02	6,0981E+01	4,1498E+06	4,8200E+06	1,4975E+01	1,2725E+03	1,0350E+03
	19	6,6949E+02	1,1094E+03	6,2670E+01	4,7841E+06	1,0102E+07	1,5332E+01	1,4542E+03	1,2030E+03
	25	7,4281E+02	1,2400E+03	6,8087E+01	5,7723E+06	2,2459E+07	1,5824E+01	1,8103E+03	1,9591E+03
	Promedio	6,0937E+02	9,6345E+02	6,0977E+01	4,1140E+06	6,6876E+06	1,5035E+01	1,3070E+03	1,1017E+03
	Desvio	6,9120E+01	1,6970E+02	3,2332E+00	9,5226E+05	4,9442E+06	3,7047E-01	1,8424E+02	3,1440E+02
3,00E+05	1	9,4951E+01	1,6420E+02	2,7275E+01	2,4289E+04	2,3519E+01	1,3232E+01	3,8752E+02	1,6053E+02
	7	1,3607E+02	2,2489E+02	2,9922E+01	7,0076E+04	6,1129E+01	1,3880E+01	4,7690E+02	2,9154E+02
	13	1,4731E+02	2,7324E+02	3,3633E+01	9,6726E+04	2,4896E+02	1,4096E+01	5,1377E+02	3,4960E+02
	19	1,8126E+02	3,1410E+02	3,4955E+01	1,2869E+05	6,3874E+02	1,4252E+01	5,8652E+02	4,1168E+02
	25	2,3462E+02	3,8109E+02	4,1950E+01	2,0714E+05	1,9478E+04	1,4461E+01	7,4439E+02	4,8257E+02
	Promedio	1,5685E+02	2,6542E+02	3,3044E+01	1,0219E+05	1,4106E+03	1,4028E+01	5,2961E+02	3,4705E+02
	Desvio	3,6764E+01	5,8735E+01	3,5304E+00	4,4259E+04	3,9522E+03	3,1226E-01	8,9342E+01	7,7661E+01

Tabla 6: Funciones de la 09 a la 16 cPSO en 30 dimensiones

FES	Problema	17	18	19	20	21	22	23	24	25
1,00E+03	1	1,1737E+03	1,4362E+03	1,3287E+03	1,2623E+03	1,5466E+03	1,9643E+03	1,6042E+03	1,5852E+03	2,0207E+03
	7	1,7355E+03	1,5290E+03	1,5052E+03	1,4980E+03	1,6758E+03	2,6841E+03	1,7484E+03	1,6789E+03	2,2788E+03
	13	1,9206E+03	1,5921E+03	1,5811E+03	1,6071E+03	1,7305E+03	3,2296E+03	1,8158E+03	1,7685E+03	2,6026E+03
	19	2,0175E+03	1,6900E+03	1,6200E+03	1,7606E+03	1,8598E+03	3,9658E+03	1,9647E+03	1,8473E+03	8,7135E+03
	25	2,7199E+03	2,0212E+03	1,8208E+03	1,9855E+03	2,5362E+03	7,6168E+03	2,3231E+03	1,9346E+03	8,0402E+04
	Promedio	1,8901E+03	1,6263E+03	1,5689E+03	1,6148E+03	1,8070E+03	3,3833E+03	1,8839E+03	1,7622E+03	1,2518E+04
	Desvio	3,2210E+02	1,4260E+02	1,1592E+02	1,8138E+02	2,4871E+02	1,1860E+03	1,9521E+02	1,1255E+02	1,9935E+04
1,00E+04	1	1,1939E+03	1,2935E+03	1,3342E+03	1,3989E+03	1,4873E+03	2,1329E+03	1,5566E+03	1,5435E+03	Nan
	7	1,5814E+03	1,4684E+03	1,4728E+03	1,5292E+03	1,6462E+03	2,7850E+03	1,7072E+03	1,7268E+03	Nan
	13	1,8519E+03	1,5718E+03	1,6733E+03	1,5899E+03	1,7977E+03	3,4899E+03	1,8231E+03	1,7812E+03	Nan
	19	2,2232E+03	1,7190E+03	1,7556E+03	1,6910E+03	1,9060E+03	4,4230E+03	1,9291E+03	1,8547E+03	Nan
	25	2,5814E+03	1,9717E+03	1,8556E+03	2,0712E+03	2,1290E+03	7,6629E+03	2,2412E+03	1,9447E+03	Nan
	Promedio	1,8673E+03	1,5921E+03	1,6221E+03	1,6285E+03	1,7934E+03	3,6809E+03	1,8453E+03	1,7731E+03	#DIV/0!
	Desvio	4,1657E+02	1,8183E+02	1,6023E+02	1,4421E+02	1,8340E+02	1,2340E+03	1,7901E+02	1,1509E+02	#DIV/0!
1,00E+05	1	1,1357E+03	1,1909E+03	1,2527E+03	1,2239E+03	1,3467E+03	1,5205E+03	1,4154E+03	1,3152E+03	Nan
	7	1,3609E+03	1,3150E+03	1,3250E+03	1,2626E+03	1,4803E+03	1,9089E+03	1,5048E+03	1,5117E+03	Nan
	13	1,4925E+03	1,3410E+03	1,3831E+03	1,3573E+03	1,5224E+03	1,9987E+03	1,5891E+03	1,5554E+03	Nan
	19	1,6370E+03	1,4176E+03	1,4187E+03	1,4408E+03	1,6286E+03	2,1376E+03	1,6882E+03	1,6564E+03	Nan
	25	1,9431E+03	1,7355E+03	1,5162E+03	1,6136E+03	1,7252E+03	4,8984E+03	1,8731E+03	1,7722E+03	Nan
	Promedio	1,5120E+03	1,3823E+03	1,3844E+03	1,3677E+03	1,5473E+03	2,2343E+03	1,5974E+03	1,5834E+03	#DIV/0!
	Desvio	2,3694E+02	1,1014E+02	7,3114E+01	1,1401E+02	9,7237E+01	7,3624E+02	1,1990E+02	1,1996E+02	#DIV/0!
3,00E+05	1	3,0900E+02	9,1518E+02	9,1060E+02	9,1502E+02	5,0720E+02	9,7183E+02	9,8698E+02	3,0236E+02	Nan
	7	4,7222E+02	9,2752E+02	9,1946E+02	9,2128E+02	5,9654E+02	1,0183E+03	1,1480E+03	5,8426E+02	Nan
	13	6,1551E+02	9,3135E+02	9,2522E+02	9,3061E+02	6,4259E+02	1,0632E+03	1,1582E+03	7,9406E+02	Nan
	19	6,7800E+02	9,4159E+02	9,4184E+02	9,4511E+02	9,4607E+02	1,0807E+03	1,2251E+03	1,1168E+03	Nan
	25	7,6480E+02	9,8231E+02	9,5678E+02	9,6872E+02	1,1844E+03	1,2102E+03	1,2843E+03	1,2308E+03	Nan
	Promedio	5,7314E+02	9,3590E+02	9,2981E+02	9,3359E+02	7,6942E+02	1,0618E+03	1,1683E+03	8,0772E+02	#DIV/0!
	Desvio	1,2607E+02	1,5388E+01	1,2884E+01	1,3649E+01	2,1391E+02	6,3702E+01	6,8073E+01	2,9949E+02	#DIV/0!

Tabla 7: Funciones de la 17 a la 25 cPSO en 30 dimensiones

Problema	1	7	13	19	25	Promedio	Desvio	Exito
01	-	-	-	-	-	-	-	0%
02	-	-	-	-	-	-	-	0%
03	-	-	-	-	-	-	-	0%
04	-	-	-	-	-	-	-	0%
05	-	-	-	-	-	-	-	0%
06	-	-	-	-	-	-	-	0%
07	-	-	-	-	-	-	-	0%
08	-	-	-	-	-	-	-	0%
09	-	-	-	-	-	-	-	0%
10	-	-	-	-	-	-	-	0%
11	-	-	-	-	-	-	-	0%
12	-	-	-	-	-	-	-	0%
13	-	-	-	-	-	-	-	0%
14	-	-	-	-	-	-	-	0%
15	-	-	-	-	-	-	-	0%
16	-	-	-	-	-	-	-	0%
17	-	-	-	-	-	-	-	0%
18	-	-	-	-	-	-	-	0%
19	-	-	-	-	-	-	-	0%
20	-	-	-	-	-	-	-	0%
21	-	-	-	-	-	-	-	0%
22	-	-	-	-	-	-	-	0%
23	-	-	-	-	-	-	-	0%
24	-	-	-	-	-	-	-	0%
25	-	-	-	-	-	-	-	0%

Tabla 8: Funciones de la 01 a la 25 cPSO en 30 dimensiones

FES	Problema	01	02	03	04	05	06	07	08
1,00E+03	1	1,1042E+02	1,0685E+02	2,8346E+06	2,0838E+03	2,9015E+03	1,6182E+05	5,4743E+02	2,1238E+01
	7	1,7146E+02	4,2362E+02	6,6665E+06	6,2090E+03	4,4953E+03	5,6244E+05	8,4017E+02	2,1590E+01
	13	2,2970E+02	6,4973E+02	1,0929E+07	9,9651E+03	5,7154E+03	1,2492E+06	1,5663E+03	2,1687E+01
	19	3,8510E+02	8,6650E+02	1,8397E+07	1,5453E+04	7,4238E+03	2,4464E+06	4,1957E+03	2,1908E+01
	25	1,3452E+03	1,7054E+03	5,1505E+07	2,6027E+04	1,0122E+04	8,3756E+06	6,7320E+03	2,1978E+01
	Promedio	3,1200E+02	6,5485E+02	1,4077E+07	1,1002E+04	5,9758E+03	1,7582E+06	2,4867E+03	2,1698E+01
	Desvio	2,5988E+02	3,5901E+02	1,0712E+07	5,9380E+03	1,9371E+03	1,8298E+06	2,0190E+03	2,2958E-01
1,00E+04	1	4,2133E+01	9,6057E+01	5,3295E+05	2,5180E+02	9,4595E+02	1,6027E+05	1,1015E+00	2,0941E+01
	7	1,3164E+02	3,2763E+02	3,9865E+06	4,7025E+02	1,3154E+03	4,3170E+05	1,2576E+00	2,1652E+01
	13	1,5820E+02	4,8576E+02	5,2288E+06	6,3277E+02	1,5922E+03	5,1444E+05	1,3596E+00	2,1741E+01
	19	2,1207E+02	6,7665E+02	6,7361E+06	9,4525E+02	1,8501E+03	7,1691E+05	1,4950E+00	2,1896E+01
	25	3,1770E+02	1,4839E+03	1,6667E+07	2,1008E+03	3,0546E+03	3,1884E+06	1,8736E+00	2,2088E+01
	Promedio	1,6936E+02	5,3952E+02	5,7569E+06	7,6556E+02	1,6497E+03	8,1668E+05	1,3908E+00	2,1727E+01
	Desvio	6,6202E+01	3,3946E+02	3,3999E+06	4,5438E+02	4,5392E+02	8,0384E+05	1,8109E-01	2,4990E-01
1,00E+05	1	0,0000E+00	0,0000E+00	4,2351E+03	3,7053E-03	1,1130E-09	4,0208E-01	3,9382E-02	2,0022E+01
	7	0,0000E+00	1,0000E-12	3,1239E+04	9,3156E-03	4,8004E-06	3,2415E+00	6,4002E-02	2,0175E+01
	13	0,0000E+00	2,0000E-12	5,4490E+04	5,7834E-01	2,1601E-04	5,5833E+00	1,2067E-01	2,0302E+01
	19	0,0000E+00	7,0000E-12	1,1646E+05	1,4080E+00	2,5663E-03	1,6205E+01	1,4782E-01	2,0580E+01
	25	3,1200E-10	1,2968E-08	2,3408E+05	1,1756E+01	4,8889E-02	3,7152E+02	4,0111E-01	2,0669E+01
	Promedio	1,3360E-11	6,0396E-10	7,5450E+04	1,7889E+00	8,0043E-03	3,7364E+01	1,3862E-01	2,0351E+01
	Desvio	6,2275E-11	2,6040E-09	6,3720E+04	3,2989E+00	1,5552E-02	8,3575E+01	1,0305E-01	2,0429E-01

Tabla 9: Funciones de la 01 a la 08 cPSO en 10 dimensiones

FES	Problema	09	10	11	12	13	14	15	16
1,00E+03	1	1,0536E+03	1,6118E+03	9,3353E+01	1,1545E+07	1,2237E+07	2,4606E+01	1,4378E+03	1,3477E+03
	7	1,2175E+03	1,8359E+03	9,6650E+01	1,4341E+07	2,0206E+07	2,4877E+01	1,6027E+03	1,5717E+03
	13	1,3013E+03	2,0900E+03	1,0268E+02	1,7190E+07	2,5763E+07	2,5136E+01	1,7101E+03	1,6902E+03
	19	1,3766E+03	2,3346E+03	1,0647E+02	1,9713E+07	3,7020E+07	2,5304E+01	1,7728E+03	1,8634E+03
	25	1,6207E+03	2,7352E+03	1,0885E+02	3,0714E+07	5,0606E+07	2,5761E+01	2,4261E+03	2,1507E+03
	Promedio	1,2954E+03	2,1077E+03	1,0147E+02	1,7900E+07	2,7669E+07	2,5117E+01	1,7269E+03	1,7083E+03
	Desvio	1,2945E+02	3,0490E+02	5,2022E+00	4,4567E+06	1,0545E+07	3,0173E-01	2,0811E+02	2,1437E+02
1,00E+04	1	1,0023E+03	1,8067E+03	9,1581E+01	1,3204E+07	9,4773E+06	2,4265E+01	1,3628E+03	1,2677E+03
	7	1,1382E+03	2,0653E+03	9,7583E+01	1,5988E+07	1,7051E+07	2,4732E+01	1,5622E+03	1,5352E+03
	13	1,2111E+03	2,2224E+03	9,8987E+01	1,8488E+07	2,3628E+07	2,5075E+01	1,7040E+03	1,6112E+03
	19	1,2881E+03	2,3590E+03	1,0184E+02	1,9795E+07	3,3852E+07	2,5208E+01	1,8085E+03	1,8663E+03
	25	1,4166E+03	2,6419E+03	1,1283E+02	2,4190E+07	4,9560E+07	2,5992E+01	1,9966E+03	2,1666E+03
	Promedio	1,2120E+03	2,2193E+03	9,9470E+01	1,8306E+07	2,5656E+07	2,5011E+01	1,6931E+03	1,6916E+03
	Desvio	1,1455E+02	2,2529E+02	4,9244E+00	2,9259E+06	1,1303E+07	4,0920E-01	1,7979E+02	2,3392E+02
1,00E+05	1	9,2086E+02	1,3910E+03	8,3937E+01	1,0128E+07	6,2207E+06	2,4281E+01	1,1206E+03	1,0536E+03
	7	1,0966E+03	1,7787E+03	9,6272E+01	1,3372E+07	1,5303E+07	2,4728E+01	1,4466E+03	1,4489E+03
	13	1,1655E+03	2,0425E+03	1,0122E+02	1,7333E+07	2,0800E+07	2,4935E+01	1,4955E+03	1,5688E+03
	19	1,2235E+03	2,1882E+03	1,0374E+02	2,0188E+07	2,9475E+07	2,5303E+01	1,5966E+03	1,6662E+03
	25	1,3936E+03	2,4306E+03	1,1032E+02	2,1721E+07	4,7490E+07	2,5664E+01	1,8236E+03	2,1284E+03
	Promedio	1,1729E+03	1,9825E+03	9,9768E+01	1,6779E+07	2,3357E+07	2,4993E+01	1,5130E+03	1,5663E+03
	Desvio	1,1209E+02	2,7304E+02	6,1786E+00	3,7809E+06	1,2032E+07	4,0256E-01	1,6396E+02	2,3061E+02
5,00E+05	1	2,6633E+02	4,4654E+02	5,6450E+01	2,2477E+05	3,9447E+02	2,2931E+01	4,6070E+02	2,3748E+02
	7	3,3240E+02	5,7065E+02	5,9987E+01	6,4798E+05	1,0202E+03	2,3474E+01	6,3899E+02	3,3676E+02
	13	3,7676E+02	6,2380E+02	6,2466E+01	8,7262E+05	2,4492E+03	2,3903E+01	6,6192E+02	4,0326E+02
	19	4,4274E+02	7,5615E+02	6,5334E+01	1,2267E+06	4,9605E+03	2,4203E+01	6,9295E+02	4,8285E+02
	25	4,9295E+02	9,5738E+02	7,0011E+01	1,5215E+06	1,3829E+05	2,4663E+01	7,8047E+02	5,4645E+02
	Promedio	3,8251E+02	6,5211E+02	6,2682E+01	9,1470E+05	1,1175E+04	2,3858E+01	6,5555E+02	4,1075E+02
	Desvio	7,2184E+01	1,3115E+02	3,7002E+00	3,5779E+05	2,8584E+04	4,5405E-01	7,1808E+01	8,9498E+01

Tabla 10: Funciones de la 09 a la 16 cPSO en 50 dimensiones

FES	Problema	17	18	19	20	21	22	23	24	25
1,00E+03	1	1,4010E+03	1,5159E+03	1,5109E+03	1,4444E+03	1,6102E+03	1,9879E+03	1,6317E+03	1,6657E+03	1,9230E+03
	7	1,8512E+03	1,6376E+03	1,6110E+03	1,5650E+03	1,7735E+03	2,2500E+03	1,7332E+03	1,8686E+03	2,1005E+03
	13	2,0636E+03	1,6808E+03	1,7174E+03	1,6217E+03	1,7976E+03	2,5269E+03	1,7998E+03	1,9436E+03	2,2743E+03
	19	2,3172E+03	1,7446E+03	1,8149E+03	1,7340E+03	1,8658E+03	4,0592E+03	1,8748E+03	2,0313E+03	2,6279E+03
	25	2,9012E+03	1,9674E+03	2,2251E+03	1,9009E+03	2,1490E+03	1,5133E+04	2,1621E+03	2,4921E+03	3,3906E+04
	Promedio	2,0924E+03	1,6975E+03	1,7327E+03	1,6460E+03	1,8266E+03	4,4668E+03	1,8117E+03	1,9662E+03	4,4343E+03
	Desvio	3,4104E+02	1,0347E+02	1,5592E+02	1,1518E+02	1,1437E+02	4,1716E+03	1,2127E+02	1,7071E+02	7,6420E+03
1,00E+04	1	1,4109E+03	1,5026E+03	1,4213E+03	1,4705E+03	1,6351E+03	1,7971E+03	1,5969E+03	1,6746E+03	Nan
	7	1,7694E+03	1,6407E+03	1,6047E+03	1,6255E+03	1,7346E+03	2,2633E+03	1,7950E+03	1,8457E+03	Nan
	13	1,9666E+03	1,6938E+03	1,7156E+03	1,6947E+03	1,7778E+03	2,9456E+03	1,8492E+03	1,9664E+03	Nan
	19	2,1271E+03	1,7917E+03	1,8195E+03	1,7894E+03	1,8714E+03	3,5577E+03	1,8883E+03	2,0407E+03	Nan
	25	3,3615E+03	1,9750E+03	1,9082E+03	2,0075E+03	1,9804E+03	2,1491E+04	1,9772E+03	2,3232E+03	Nan
	Promedio	2,0548E+03	1,7046E+03	1,7013E+03	1,7035E+03	1,7964E+03	4,2519E+03	1,8329E+03	1,9726E+03	#DIV/0!
	Desvio	4,2852E+02	1,1299E+02	1,3362E+02	1,2730E+02	8,6686E+01	4,6688E+03	8,6430E+01	1,6019E+02	#DIV/0!
1,00E+05	1	1,4196E+03	1,4393E+03	1,4324E+03	1,3417E+03	1,5355E+03	1,9105E+03	1,5762E+03	1,6567E+03	Nan
	7	1,7113E+03	1,5252E+03	1,5409E+03	1,5235E+03	1,6585E+03	2,0871E+03	1,7078E+03	1,7978E+03	Nan
	13	1,8363E+03	1,6066E+03	1,5840E+03	1,5716E+03	1,7296E+03	2,2662E+03	1,7531E+03	1,8904E+03	Nan
	19	2,0054E+03	1,6630E+03	1,6953E+03	1,6700E+03	1,7739E+03	3,4413E+03	1,8065E+03	1,9510E+03	Nan
	25	2,1678E+03	1,8762E+03	1,8540E+03	1,8887E+03	1,8694E+03	9,3032E+03	1,9383E+03	2,1320E+03	Nan
	Promedio	1,8512E+03	1,6165E+03	1,6213E+03	1,5905E+03	1,7182E+03	3,0363E+03	1,7536E+03	1,8759E+03	#DIV/0!
	Desvio	1,7651E+02	1,2575E+02	1,1270E+02	1,1994E+02	8,8722E+01	1,6899E+03	9,2844E+01	1,1512E+02	#DIV/0!
5,00E+05	1	5,4668E+02	9,5523E+02	9,6200E+02	9,5678E+02	7,7823E+02	1,0371E+03	1,0547E+03	1,2042E+03	Nan
	7	6,7647E+02	9,8506E+02	9,9660E+02	9,9464E+02	1,0757E+03	1,1479E+03	1,1044E+03	1,2632E+03	Nan
	13	7,6460E+02	1,0012E+03	1,0073E+03	1,0049E+03	1,0914E+03	1,1686E+03	1,1195E+03	1,3225E+03	Nan
	19	8,5094E+02	1,0233E+03	1,0322E+03	1,0398E+03	1,1278E+03	1,1879E+03	1,1542E+03	1,3529E+03	Nan
	25	1,1683E+03	1,0633E+03	1,0749E+03	1,0745E+03	1,2645E+03	1,3523E+03	1,2874E+03	1,3949E+03	Nan
	Promedio	7,8601E+02	1,0040E+03	1,0123E+03	1,0134E+03	1,0901E+03	1,1721E+03	1,1348E+03	1,3094E+03	#DIV/0!
	Desvio	1,6356E+02	2,6543E+01	2,7212E+01	3,2364E+01	9,2810E+01	6,1514E+01	5,5426E+01	5,5623E+01	#DIV/0!

Tabla 11: Funciones de la 17 a la 25 cPSO en 50 dimensiones

Problema	1	7	13	19	25	Promedio	Desvio	Exito
01	-	-	-	-	-	-	-	0%
02	-	-	-	-	-	-	-	0%
03	-	-	-	-	-	-	-	0%
04	-	-	-	-	-	-	-	0%
05	-	-	-	-	-	-	-	0%
06	-	-	-	-	-	-	-	0%
07	-	-	-	-	-	-	-	0%
08	-	-	-	-	-	-	-	0%
09	-	-	-	-	-	-	-	0%
10	-	-	-	-	-	-	-	0%
11	-	-	-	-	-	-	-	0%
12	-	-	-	-	-	-	-	0%
13	-	-	-	-	-	-	-	0%
14	-	-	-	-	-	-	-	0%
15	-	-	-	-	-	-	-	0%
16	-	-	-	-	-	-	-	0%
17	-	-	-	-	-	-	-	0%
18	-	-	-	-	-	-	-	0%
19	-	-	-	-	-	-	-	0%
20	-	-	-	-	-	-	-	0%
21	-	-	-	-	-	-	-	0%
22	-	-	-	-	-	-	-	0%
23	-	-	-	-	-	-	-	0%
24	-	-	-	-	-	-	-	0%
25	-	-	-	-	-	-	-	0%

Tabla 12: Funciones de la 01 a la 25 cPSO en 50 dimensiones

FES	Problema	01	02	03	04	05	06	07	08
1,00E+03	1	1,7031E+03	2,3211E+03	1,9102E+07	1,1679E+04	8,6922E+03	4,7027E+06	1,0544E+03	2,0869E+01
	7	2,6506E+03	7,5362E+03	6,7246E+07	1,8856E+04	1,3676E+04	6,4237E+07	1,6160E+03	2,1558E+01
	13	4,5920E+03	9,5917E+03	1,1388E+08	2,5430E+04	1,4512E+04	1,6081E+08	2,9696E+03	2,1676E+01
	19	5,8117E+03	1,2306E+04	1,7264E+08	3,5098E+04	1,8046E+04	5,1271E+08	4,4018E+03	2,1791E+01
	25	9,8560E+03	1,9684E+04	3,1113E+08	4,0447E+04	2,0911E+04	9,7900E+08	7,1623E+03	2,2095E+01
	Promedio	4,6393E+03	9,8519E+03	1,2899E+08	2,6242E+04	1,5189E+04	3,0891E+08	3,3767E+03	2,1634E+01
	Desvio	2,2292E+03	4,1640E+03	8,2919E+07	9,3098E+03	3,1365E+03	3,2525E+08	1,9507E+03	2,6045E-01
1,00E+04	1	1,1151E+02	3,2803E+02	3,0791E+06	9,1868E+02	1,4799E+03	1,6652E+05	1,2488E+00	2,1285E+01
	7	2,4830E+02	5,8211E+02	8,9065E+06	1,9199E+03	2,4795E+03	9,9076E+05	1,3991E+00	2,1569E+01
	13	3,6977E+02	8,9746E+02	1,5532E+07	3,3055E+03	2,8082E+03	1,1240E+06	1,5770E+00	2,1800E+01
	19	4,7136E+02	1,1735E+03	1,8432E+07	5,4016E+03	3,3603E+03	1,8266E+06	2,0118E+00	2,1891E+01
	25	1,1091E+03	3,1169E+03	7,0608E+07	1,8645E+04	4,4184E+03	8,9191E+06	2,5402E+00	2,2120E+01
	Promedio	4,0746E+02	1,0204E+03	1,8499E+07	4,7822E+03	2,8470E+03	1,9768E+06	1,7085E+00	2,1750E+01
	Desvio	2,3626E+02	6,5717E+02	1,7152E+07	4,4337E+03	6,9845E+02	2,2263E+06	3,8665E-01	2,0178E-01
1,00E+05	1	0,0000E+00	0,0000E+00	2,9811E+04	1,2087E-03	1,9709E-06	4,5149E-02	8,9317E-04	2,1052E+01
	7	0,0000E+00	2,0000E-12	6,9935E+05	9,8177E-03	1,6735E-04	3,8479E+00	4,6737E-02	2,1538E+01
	13	0,0000E+00	1,5900E-10	1,9111E+06	1,6946E-01	8,1875E-04	7,5050E+00	7,1126E-02	2,1739E+01
	19	0,0000E+00	7,0225E-07	3,5917E+06	1,1837E+02	1,1118E-01	9,4149E+03	1,0259E+00	2,1908E+01
	25	2,0300E-09	4,4691E+02	8,6630E+06	1,7001E+04	1,3481E+03	6,1444E+05	1,0791E+00	2,2086E+01
	Promedio	8,1240E-11	5,5354E+01	2,5353E+06	7,8684E+02	9,7295E+01	5,1254E+04	3,9705E-01	2,1714E+01
	Desvio	4,0599E-10	1,2942E+02	2,1921E+06	3,3859E+03	3,3218E+02	1,3987E+05	4,7310E-01	2,8068E-01

Tabla 13: Funciones de la 01 a la 08 dPSO en 10 dimensiones

FES	Problema	09	10	11	12	13	14	15	16
1,00E+03	1	1,3144E+02	1,4310E+02	1,8810E+01	1,3156E+05	5,9904E+03	4,6275E+00	1,1818E+03	7,2846E+02
	7	1,9884E+02	2,4038E+02	1,9696E+01	5,5340E+05	7,8971E+05	4,9451E+00	1,4073E+03	1,1082E+03
	13	2,2690E+02	3,3405E+02	1,9840E+01	8,1524E+05	3,0457E+06	5,0229E+00	1,5566E+03	1,4356E+03
	19	2,6006E+02	4,2742E+02	2,3814E+01	9,8319E+05	1,0294E+07	5,0702E+00	1,7039E+03	1,8951E+03
	25	3,5077E+02	5,5842E+02	2,3814E+01	1,3219E+06	2,0627E+07	5,4012E+00	2,4876E+03	2,5604E+03
	Promedio	2,3395E+02	3,2605E+02	2,1625E+01	7,7824E+05	5,8076E+06	5,0244E+00	1,5937E+03	1,5103E+03
	Desvio	5,1574E+01	1,1845E+02	2,1547E+00	2,8345E+05	6,4867E+06	1,6391E-01	2,8473E+02	4,9149E+02
1,00E+04	1	1,1636E+02	1,1120E+02	1,3803E+01	2,9444E+05	1,2534E+04	4,5343E+00	8,3935E+02	5,6662E+02
	7	1,6983E+02	2,0496E+02	1,6413E+01	5,7062E+05	2,0261E+05	4,7495E+00	1,1734E+03	8,4538E+02
	13	1,9709E+02	2,5725E+02	1,6924E+01	7,7067E+05	1,1584E+06	4,9383E+00	1,3012E+03	1,2601E+03
	19	2,1168E+02	3,0330E+02	1,9276E+01	8,7657E+05	4,6092E+06	5,1223E+00	1,6281E+03	1,6367E+03
	25	2,8668E+02	4,0761E+02	2,1009E+01	1,2750E+06	1,0167E+07	5,4222E+00	2,2556E+03	2,1802E+03
	Promedio	1,9264E+02	2,5205E+02	1,7367E+01	7,3869E+05	2,7243E+06	4,9770E+00	1,3705E+03	1,3010E+03
	Desvio	3,9296E+01	7,7602E+01	1,9070E+00	2,4866E+05	3,2223E+06	2,5324E-01	3,2204E+02	4,9245E+02
1,00E+05	1	3,9798E+00	9,9496E+00	5,7185E+00	4,9102E+00	4,7690E-01	4,4944E+00	2,0645E+02	1,2050E+02
	7	1,3930E+01	2,6872E+01	9,6629E+00	3,8319E+01	2,3979E+00	4,6100E+00	4,2590E+02	1,6394E+02
	13	2,0894E+01	4,4827E+01	1,7888E+01	4,2254E+02	4,5519E+00	4,7656E+00	4,4609E+02	1,6993E+02
	19	2,8854E+01	1,5029E+02	1,9910E+01	2,3845E+03	1,3622E+02	5,1548E+00	5,1096E+02	2,1185E+02
	25	1,5535E+02	2,1909E+02	2,5012E+01	1,0997E+04	1,1038E+05	5,3769E+00	9,8878E+02	6,7309E+02
	Promedio	2,9078E+01	7,3910E+01	1,5677E+01	1,8130E+03	8,9335E+03	4,8793E+00	5,2584E+02	2,1850E+02
	Desvio	3,2033E+01	6,6156E+01	6,2074E+00	2,7304E+03	2,9628E+04	2,8251E-01	2,0513E+02	1,2584E+02

Tabla 14: Funciones de la 09 a la 16 dPSO en 10 dimensiones

FES	Problema	17	18	19	20	21	22	23	24	25
1,00E+03	1	6,7802E+02	1,2948E+03	1,3083E+03	1,2781E+03	1,5870E+03	1,3968E+03	1,4816E+03	1,3332E+03	1,6963E+03
	7	1,3094E+03	1,9143E+03	1,6808E+03	1,7696E+03	1,7876E+03	2,0719E+03	1,6187E+03	1,5426E+03	2,0518E+03
	13	1,6511E+03	2,0701E+03	1,9525E+03	2,0597E+03	1,9914E+03	2,7161E+03	1,7390E+03	1,8410E+03	2,3782E+03
	19	2,0732E+03	2,4174E+03	2,3446E+03	2,4365E+03	2,1286E+03	3,3738E+03	1,8995E+03	1,9189E+03	4,1499E+03
	25	2,9868E+03	3,5545E+03	3,8082E+03	3,4431E+03	2,6253E+03	1,9947E+05	2,7562E+03	2,2366E+03	1,0308E+06
	Promedio	1,6876E+03	2,1530E+03	2,1377E+03	2,0974E+03	1,9708E+03	1,3584E+04	1,8186E+03	1,7715E+03	4,9112E+04
	Desvio	5,5128E+02	5,3551E+02	6,7164E+02	5,3431E+02	2,4487E+02	3,9640E+04	2,7782E+02	2,2951E+02	2,0520E+05
1,00E+04	1	6,4823E+02	1,1927E+03	1,2287E+03	1,2691E+03	1,5569E+03	1,3114E+03	1,5097E+03	1,4810E+03	1,6890E+03
	7	8,8434E+02	1,5454E+03	1,4254E+03	1,5660E+03	1,6155E+03	1,8632E+03	1,6523E+03	1,5879E+03	2,0135E+03
	13	1,2612E+03	1,6175E+03	1,6563E+03	1,7636E+03	1,7263E+03	2,5545E+03	1,6912E+03	1,6637E+03	2,1645E+03
	19	1,9088E+03	1,9685E+03	1,9372E+03	2,1297E+03	1,7830E+03	3,0087E+03	1,8266E+03	1,7733E+03	NaN
	25	2,5005E+03	3,9649E+03	2,6108E+03	3,4598E+03	2,5289E+03	7,8129E+04	2,1511E+03	2,0479E+03	NaN
	Promedio	1,4201E+03	1,8071E+03	1,7466E+03	1,9293E+03	1,7504E+03	9,1280E+03	1,7445E+03	1,6846E+03	2,1715E+03
	Desvio	6,0919E+02	5,5037E+02	3,8484E+02	5,4205E+02	2,0353E+02	2,0493E+04	1,6391E+02	1,3377E+02	4,6650E+02
1,00E+05	1	2,5747E+02	7,3133E+02	4,4141E+02	4,1604E+02	5,0000E+02	7,7980E+02	5,5947E+02	2,0000E+02	1,0667E+03
	7	4,8443E+02	8,0002E+02	8,0011E+02	8,0010E+02	5,0000E+02	7,9991E+02	5,5947E+02	2,0000E+02	1,8817E+03
	13	5,1471E+02	9,8034E+02	9,5260E+02	9,4820E+02	1,0195E+03	8,4134E+02	1,2303E+03	2,0000E+02	2,3883E+03
	19	5,9629E+02	1,0515E+03	1,1778E+03	1,0188E+03	1,1578E+03	1,0641E+03	1,2610E+03	5,0000E+02	NaN
	25	6,9663E+02	1,3714E+03	1,4282E+03	1,3874E+03	1,4443E+03	1,8285E+03	1,3850E+03	1,5047E+03	NaN
	Promedio	5,3394E+02	9,9184E+02	9,7584E+02	9,2373E+02	8,6898E+02	9,4718E+02	9,8939E+02	4,4116E+02	2,0047E+03
	Desvio	9,7586E+01	1,9594E+02	2,4308E+02	1,9334E+02	3,4729E+02	2,4050E+02	3,3810E+02	4,1653E+02	4,0077E+02

Tabla 15: Funciones de la 17 a la 25 dPSO en 10 dimensiones

Problema	1	7	13	19	25	Promedio	Desvio	Exito
01	9,6024E+04	6,9274E+04	6,5772E+04	6,3521E+04	5,6775E+04	6,7940E+04	8,6829E+03	100%
02	9,8271E+04	8,5774E+04	7,5524E+04	6,8274E+04	0,0000E+00	6,1368E+04	3,6178E+04	76%
03	-	-	-	-	-	-	-	0%
04	-	-	-	-	-	-	-	0%
05	-	-	-	-	-	-	-	0%
06	-	-	-	-	-	-	-	0%
07	-	-	-	-	-	-	-	0%
08	-	-	-	-	-	-	-	0%
09	-	-	-	-	-	-	-	0%
10	-	-	-	-	-	-	-	0%
11	-	-	-	-	-	-	-	0%
12	-	-	-	-	-	-	-	0%
13	-	-	-	-	-	-	-	0%
14	-	-	-	-	-	-	-	0%
15	-	-	-	-	-	-	-	0%
16	-	-	-	-	-	-	-	0%
17	-	-	-	-	-	-	-	0%
18	-	-	-	-	-	-	-	0%
19	-	-	-	-	-	-	-	0%
20	-	-	-	-	-	-	-	0%
21	-	-	-	-	-	-	-	0%
22	-	-	-	-	-	-	-	0%
23	-	-	-	-	-	-	-	0%
24	-	-	-	-	-	-	-	0%
25	-	-	-	-	-	-	-	0%

Tabla 16: Funciones de la 01 a la 25 dPSO en 10 dimensiones

FES	Problema	01	02	03	04	05	06	07	08
1,00E+03	1	1,6189E+04	3,5138E+04	2,5905E+08	6,9005E+04	3,7504E+04	4,5276E+09	3,7495E+03	2,1258E+01
	7	3,9318E+04	4,6553E+04	5,1567E+08	9,7338E+04	4,1108E+04	1,1982E+10	5,8826E+03	2,1609E+01
	13	4,4993E+04	5,2255E+04	7,6076E+08	1,1434E+05	4,7433E+04	1,7640E+10	9,1052E+03	2,1688E+01
	19	5,0650E+04	6,1793E+04	9,0656E+08	2,0118E+05	4,9908E+04	2,0395E+10	1,4449E+04	2,1765E+01
	25	6,4413E+04	9,7711E+04	1,3087E+09	5,3294E+05	5,9554E+04	3,9955E+10	2,0196E+04	2,1844E+01
	Promedio	4,4202E+04	5,6661E+04	7,6026E+08	1,5817E+05	4,6889E+04	1,8527E+10	1,0166E+04	2,1655E+01
	Desvio	1,0372E+04	1,5198E+04	3,1179E+08	9,8585E+04	6,2180E+03	9,0519E+09	5,2311E+03	1,6116E-01
1,00E+04	1	2,4739E+03	1,4073E+04	7,7084E+07	5,0387E+04	1,0826E+04	3,4917E+07	2,0021E+01	2,1060E+01
	7	3,8697E+03	2,1504E+04	1,0777E+08	7,2276E+04	1,2612E+04	1,1657E+08	4,8419E+01	2,1634E+01
	13	4,3326E+03	2,6571E+04	1,5509E+08	8,3001E+04	1,4145E+04	1,4231E+08	5,8530E+01	2,1730E+01
	19	5,9068E+03	3,0473E+04	2,0761E+08	1,0626E+05	1,5498E+04	2,5266E+08	8,6879E+01	2,1825E+01
	25	9,3244E+03	4,3283E+04	3,0392E+08	2,4946E+05	1,7555E+04	9,3140E+08	2,9138E+02	2,1906E+01
	Promedio	5,0539E+03	2,7030E+04	1,5957E+08	9,3974E+04	1,3973E+04	2,3787E+08	8,4686E+01	2,1699E+01
	Desvio	1,8197E+03	7,8680E+03	5,5377E+07	3,9832E+04	1,9419E+03	2,3987E+08	6,6415E+01	1,9190E-01
1,00E+05	1	3,5577E+02	3,1614E+03	1,8336E+07	3,5461E+04	5,6762E+03	2,0749E+06	1,4970E+00	2,1515E+01
	7	5,1621E+02	5,4877E+03	4,3562E+07	5,0257E+04	6,7475E+03	3,8670E+06	1,7201E+00	2,1665E+01
	13	6,9989E+02	8,2133E+03	5,4172E+07	6,2720E+04	7,6031E+03	5,1339E+06	1,9726E+00	2,1736E+01
	19	8,4310E+02	1,2764E+04	6,8841E+07	8,1718E+04	8,2132E+03	7,5940E+06	2,1392E+00	2,1767E+01
	25	1,7471E+03	1,8744E+04	9,3646E+07	1,0027E+05	1,0146E+04	3,1568E+07	3,2131E+00	2,1901E+01
	Promedio	7,3286E+02	9,0654E+03	5,5454E+07	6,5348E+04	7,5260E+03	7,9592E+06	2,0031E+00	2,1723E+01
	Desvio	2,8349E+02	4,2919E+03	1,8545E+07	1,8818E+04	1,0896E+03	7,7180E+06	3,9669E-01	8,1740E-02
3,00E+05	1	0,0000E+00	1,8184E-02	5,8120E+05	2,8514E+04	1,9518E+03	2,5903E-02	2,0808E-06	2,1273E+01
	7	0,0000E+00	1,8269E+03	2,6513E+06	4,0340E+04	3,7673E+03	2,3389E+01	9,8649E-03	2,1390E+01
	13	0,0000E+00	5,8100E+03	4,1367E+07	4,3365E+04	5,0303E+03	1,5770E+02	2,9498E-02	2,1472E+01
	19	4,7424E+02	9,6245E+03	5,3906E+07	5,1561E+04	7,0499E+03	1,0502E+06	1,6808E+00	2,1503E+01
	25	9,1141E+02	1,7408E+04	9,5359E+07	6,6421E+04	9,7820E+03	1,7382E+07	2,7332E+00	2,1623E+01
	Promedio	2,3825E+02	5,7709E+03	3,4440E+07	4,6315E+04	5,3382E+03	1,8805E+06	6,9206E-01	2,1450E+01
	Desvio	3,1878E+02	4,8821E+03	2,7838E+07	1,0452E+04	2,1203E+03	4,6074E+06	9,5141E-01	9,2910E-02

Tabla 17: Funciones de la 01 a la 08 dPSO en 30 dimensiones

FES	Problema	09	10	11	12	13	14	15	16
1,00E+03	1	5,7125E+02	7,8420E+02	4,8434E+01	2,4322E+06	3,2951E+06	1,4612E+01	1,3096E+03	1,3338E+03
	7	7,0119E+02	1,1685E+03	5,3112E+01	3,7556E+06	9,1132E+06	1,4946E+01	1,5261E+03	1,6477E+03
	13	7,5325E+02	1,3712E+03	5,7902E+01	4,8465E+06	1,0997E+07	1,4981E+01	1,7057E+03	1,7843E+03
	19	7,9091E+02	1,6419E+03	5,9904E+01	5,3711E+06	2,1263E+07	1,5125E+01	1,8283E+03	1,9254E+03
	25	9,8702E+02	2,2402E+03	7,1950E+01	6,2938E+06	4,5527E+07	1,5622E+01	2,5891E+03	2,1416E+03
	Promedio	7,4801E+02	1,4058E+03	5,7892E+01	4,6042E+06	1,6314E+07	1,5035E+01	1,7095E+03	1,7737E+03
	Desvio	9,6642E+01	3,6239E+02	5,4663E+00	1,0453E+06	1,0259E+07	2,2569E-01	2,7954E+02	2,2979E+02
1,00E+04	1	5,3646E+02	8,4979E+02	4,6517E+01	3,1321E+06	2,7437E+04	1,4504E+01	1,2001E+03	1,2240E+03
	7	6,2458E+02	1,0260E+03	5,3112E+01	4,1823E+06	3,6389E+06	1,4796E+01	1,3992E+03	1,3964E+03
	13	6,8641E+02	1,2895E+03	5,3112E+01	4,8449E+06	7,0792E+06	1,4980E+01	1,5375E+03	1,6841E+03
	19	7,2515E+02	1,4922E+03	5,3112E+01	5,2374E+06	1,1198E+07	1,5049E+01	1,6954E+03	1,8061E+03
	25	9,4018E+02	1,9921E+03	6,8063E+01	7,7815E+06	3,2514E+07	1,5462E+01	2,0913E+03	2,1763E+03
	Promedio	6,9271E+02	1,2668E+03	5,4351E+01	4,8734E+06	8,3268E+06	1,4954E+01	1,5647E+03	1,6243E+03
	Desvio	1,0520E+02	2,8903E+02	4,0279E+00	9,7464E+05	7,0733E+06	2,4936E-01	2,1765E+02	2,6293E+02
1,00E+05	1	4,2633E+02	6,2283E+02	5,3231E+01	3,2338E+06	2,3969E+02	1,4526E+01	1,0866E+03	7,2420E+02
	7	5,2876E+02	8,0913E+02	5,8158E+01	4,4796E+06	1,3291E+06	1,4748E+01	1,2643E+03	1,0991E+03
	13	5,7045E+02	9,1385E+02	6,1870E+01	5,1531E+06	3,5262E+06	1,4925E+01	1,2954E+03	1,1470E+03
	19	6,2225E+02	1,0508E+03	6,4522E+01	5,8535E+06	7,1175E+06	1,5163E+01	1,4023E+03	1,3726E+03
	25	6,8790E+02	1,6354E+03	6,7550E+01	7,1785E+06	1,3426E+07	1,5646E+01	1,8295E+03	1,8766E+03
	Promedio	5,7146E+02	9,5574E+02	6,1114E+01	5,1366E+06	4,4540E+06	1,4954E+01	1,3426E+03	1,2306E+03
	Desvio	6,1756E+01	2,1515E+02	4,3343E+00	1,0203E+06	3,7934E+06	3,0733E-01	1,5903E+02	2,7150E+02
3,00E+05	1	7,3627E+01	1,1921E+02	2,8348E+01	1,9579E+03	7,5948E+00	1,3690E+01	3,0270E+02	1,3345E+02
	7	9,1536E+01	1,9203E+02	4,7575E+01	9,7304E+03	1,3488E+02	1,4300E+01	4,0435E+02	3,1348E+02
	13	1,1840E+02	2,1988E+02	4,9838E+01	1,6545E+04	4,4646E+05	1,4430E+01	5,3528E+02	4,0127E+02
	19	1,4725E+02	4,8192E+02	5,2276E+01	6,2219E+04	8,4374E+05	1,4554E+01	1,1405E+03	5,0000E+02
	25	3,7800E+02	7,8685E+02	5,5165E+01	2,7574E+05	2,3356E+06	1,4652E+01	1,3468E+03	1,2389E+03
	Promedio	1,3477E+02	3,2886E+02	4,8222E+01	4,0303E+04	5,2421E+05	1,4391E+01	7,3009E+02	4,8568E+02
	Desvio	7,0166E+01	2,2136E+02	6,4462E+00	5,7001E+04	6,0272E+05	2,1681E-01	3,7684E+02	2,6995E+02

Tabla 18: Funciones de la 09 a la 16 dPSO en 30 dimensiones

FES	Problema	17	18	19	20	21	22	23	24	25
1,00E+03	1	1,4619E+03	1,4089E+03	1,4636E+03	1,4654E+03	1,5376E+03	2,1608E+03	1,4954E+03	1,7416E+03	2,1197E+03
	7	1,9185E+03	1,5821E+03	1,5424E+03	1,6085E+03	1,8209E+03	2,8317E+03	1,7738E+03	2,3293E+03	2,4824E+03
	13	2,1873E+03	1,6673E+03	1,6845E+03	1,6578E+03	1,8778E+03	3,5401E+03	1,8563E+03	1,8256E+03	2,5991E+03
	19	2,3948E+03	1,8435E+03	1,7404E+03	1,7555E+03	1,9943E+03	4,4692E+03	2,0515E+03	1,9702E+03	2,6309E+03
	25	2,5075E+03	1,9859E+03	2,1162E+03	2,0926E+03	2,3190E+03	1,5615E+04	2,5935E+03	2,0708E+03	1,4762E+04
	Promedio	2,1506E+03	1,7070E+03	1,6877E+03	1,6837E+03	1,8997E+03	4,2745E+03	1,9255E+03	1,8283E+03	3,2724E+03
	Desvio	2,8801E+02	1,6799E+02	1,7986E+02	1,2667E+02	1,7636E+02	2,7729E+03	2,6099E+02	1,5746E+02	2,5779E+03
1,00E+04	1	1,6699E+03	1,3927E+03	1,4030E+03	9,1096E+02	1,5423E+03	1,9120E+03	1,6363E+03	1,6379E+03	2,0360E+03
	7	1,9373E+03	1,5305E+03	1,5535E+03	1,5838E+03	1,7914E+03	2,6397E+03	1,7579E+03	1,8560E+03	2,1005E+03
	13	2,1298E+03	1,5870E+03	1,6582E+03	1,6361E+03	1,9284E+03	3,0752E+03	1,8554E+03	1,7830E+03	2,1979E+03
	19	2,4300E+03	1,7491E+03	1,8039E+03	1,7122E+03	2,0759E+03	3,4493E+03	1,9593E+03	1,9929E+03	2,4167E+03
	25	3,0404E+03	2,2792E+03	2,1154E+03	1,9315E+03	2,9120E+03	5,1697E+03	2,6056E+03	1,9052E+03	Nan
	Promedio	2,2035E+03	1,6577E+03	1,6781E+03	1,6313E+03	1,9724E+03	3,2035E+03	1,9023E+03	1,8084E+03	2,2494E+03
	Desvio	3,6139E+02	1,8791E+02	2,0009E+02	1,8836E+02	2,9119E+02	8,3485E+02	2,3157E+02	1,0944E+02	2,0136E+02
1,00E+05	1	1,0461E+03	1,2878E+03	1,2738E+03	9,2196E+02	1,3567E+03	1,6952E+03	1,4182E+03	1,4746E+03	2,0682E+03
	7	1,3149E+03	1,3708E+03	1,3762E+03	1,3672E+03	1,4604E+03	1,9582E+03	1,5122E+03	1,6277E+03	2,1005E+03
	13	1,5663E+03	1,4220E+03	1,4402E+03	1,4937E+03	1,5711E+03	2,2568E+03	1,6384E+03	1,8761E+03	2,2914E+03
	19	1,8208E+03	1,4785E+03	1,5194E+03	1,5870E+03	1,6638E+03	2,6207E+03	1,6962E+03	1,5976E+03	2,5800E+03
	25	2,1993E+03	1,7275E+03	1,7939E+03	1,7482E+03	1,8591E+03	3,3634E+03	2,1353E+03	1,7255E+03	Nan
	Promedio	1,5672E+03	1,4390E+03	1,4697E+03	1,4739E+03	1,5742E+03	2,3548E+03	1,6350E+03	1,6269E+03	2,2379E+03
	Desvio	2,9961E+02	1,0818E+02	1,1757E+02	1,7761E+02	1,3140E+02	5,1892E+02	1,6562E+02	1,1068E+02	1,6762E+02
3,00E+05	1	4,2556E+02	9,1444E+02	9,1885E+02	9,0406E+02	5,0000E+02	9,9403E+02	5,3749E+02	2,0000E+02	1,9099E+03
	7	5,9132E+02	9,3434E+02	9,3075E+02	9,2559E+02	5,0000E+02	1,0505E+03	7,1336E+02	2,0000E+02	2,0974E+03
	13	7,4001E+02	1,1321E+03	9,4430E+02	9,4872E+02	5,0000E+02	1,1059E+03	1,1316E+03	1,5720E+03	2,1005E+03
	19	9,5242E+02	1,2756E+03	1,2270E+03	1,2764E+03	7,5923E+02	1,5779E+03	1,2086E+03	2,0000E+02	2,2832E+03
	25	1,2389E+03	1,3740E+03	1,4270E+03	1,4847E+03	1,4137E+03	1,9220E+03	1,4707E+03	2,0000E+02	Nan
	Promedio	7,8647E+02	1,1164E+03	1,0540E+03	1,1016E+03	6,9525E+02	1,2704E+03	1,0122E+03	5,5881E+02	2,0932E+03
	Desvio	2,3397E+02	1,7300E+02	1,8092E+02	2,0446E+02	3,3637E+02	3,0674E+02	2,7582E+02	5,8832E+02	9,9897E+01

Tabla 19: Funciones de la 17 a la 25 dPSO en 30 dimensiones

Problema	1	7	13	19	25	Promedio	Desvio	Exito
01	2,5580E+05	2,1302E+05	1,9852E+05	0,0000E+00	0,0000E+00	1,1910E+05	1,0843E+05	56%
02	-	-	-	-	-	-	-	0%
03	-	-	-	-	-	-	-	0%
04	-	-	-	-	-	-	-	0%
05	-	-	-	-	-	-	-	0%
06	-	-	-	-	-	-	-	0%
07	-	-	-	-	-	-	-	0%
08	-	-	-	-	-	-	-	0%
09	-	-	-	-	-	-	-	0%
10	-	-	-	-	-	-	-	0%
11	-	-	-	-	-	-	-	0%
12	-	-	-	-	-	-	-	0%
13	-	-	-	-	-	-	-	0%
14	-	-	-	-	-	-	-	0%
15	-	-	-	-	-	-	-	0%
16	-	-	-	-	-	-	-	0%
17	-	-	-	-	-	-	-	0%
18	-	-	-	-	-	-	-	0%
19	-	-	-	-	-	-	-	0%
20	-	-	-	-	-	-	-	0%
21	-	-	-	-	-	-	-	0%
22	-	-	-	-	-	-	-	0%
23	-	-	-	-	-	-	-	0%
24	-	-	-	-	-	-	-	0%
25	-	-	-	-	-	-	-	0%

Tabla 20: Funciones de la 01 a la 25 dPSO en 30 dimensiones

FES	Problema	01	02	03	04	05	06	07	08
1,00E+03	1	6,7716E+04	9,1182E+04	8,8396E+08	1,7458E+05	3,6817E+04	1,5095E+10	4,9865E+03	2,1556E+01
	7	9,1570E+04	1,1057E+05	3,1043E+09	2,3405E+05	4,4160E+04	3,2804E+10	8,0357E+03	2,1670E+01
	13	1,0513E+05	1,2616E+05	4,2392E+09	3,0044E+05	4,6096E+04	4,4855E+10	1,1008E+04	2,1687E+01
	19	1,0863E+05	1,4247E+05	5,6120E+09	4,4907E+05	5,1753E+04	5,0446E+10	1,8419E+04	2,1772E+01
	25	1,4054E+05	2,4311E+05	8,2133E+09	7,1957E+05	5,7662E+04	7,1368E+10	2,5163E+04	2,1913E+01
	Promedio	1,0197E+05	1,3185E+05	4,2199E+09	3,4038E+05	4,7448E+04	4,2234E+10	1,2957E+04	2,1710E+01
	Desvio	1,5066E+04	3,4034E+04	1,9008E+09	1,4079E+05	4,8803E+03	1,3200E+10	6,2080E+03	8,7557E-02
1,00E+04	1	6,8544E+03	4,6688E+04	1,9772E+08	1,2057E+05	2,1399E+04	2,9113E+08	1,8328E+02	2,1536E+01
	7	1,1672E+04	7,7435E+04	3,8445E+08	1,6607E+05	2,3249E+04	6,2496E+08	2,4271E+02	2,1652E+01
	13	1,2986E+04	8,3566E+04	4,5171E+08	2,0353E+05	2,4500E+04	8,5238E+08	2,9225E+02	2,1713E+01
	19	1,7247E+04	9,8633E+04	5,6437E+08	2,7733E+05	2,6917E+04	1,4136E+09	3,9526E+02	2,1798E+01
	25	2,7523E+04	1,7837E+05	8,6189E+08	3,5463E+05	3,3059E+04	2,5748E+09	7,2711E+02	2,1872E+01
	Promedio	1,4599E+04	9,0052E+04	4,7441E+08	2,2160E+05	2,5418E+04	1,1377E+09	3,2860E+02	2,1714E+01
	Desvio	5,1176E+03	2,9422E+04	1,5453E+08	6,9032E+04	2,8629E+03	7,0070E+08	1,3053E+02	1,0237E-01
1,00E+05	1	1,6307E+03	2,4348E+04	9,3986E+07	7,7677E+04	1,2253E+04	7,2528E+06	3,3400E+00	2,1511E+01
	7	2,4797E+03	4,3786E+04	1,4707E+08	1,1822E+05	1,4210E+04	3,2394E+07	4,2261E+00	2,1636E+01
	13	2,8502E+03	4,9006E+04	1,7849E+08	1,4197E+05	1,5809E+04	4,5362E+07	4,5023E+00	2,1681E+01
	19	3,5755E+03	5,4488E+04	2,2249E+08	1,8682E+05	1,8615E+04	7,3939E+07	6,1627E+00	2,1790E+01
	25	4,1262E+03	8,5408E+04	3,3076E+08	2,4362E+05	2,1884E+04	2,1959E+08	1,0978E+01	2,1890E+01
	Promedio	2,9163E+03	5,0199E+04	1,9212E+08	1,5171E+05	1,6281E+04	6,0715E+07	5,1089E+00	2,1695E+01
	Desvio	6,6719E+02	1,1941E+04	6,0718E+07	4,6120E+04	2,5092E+03	4,8813E+07	1,6779E+00	1,0682E-01
5,00E+05	1	0,0000E+00	1,9144E+04	2,2837E+06	4,4357E+04	4,1838E+03	3,5649E+01	6,7374E-04	2,1411E+01
	7	1,2287E+03	3,3922E+04	5,9728E+07	1,0154E+05	8,6604E+03	2,0524E+06	1,6792E+00	2,1494E+01
	13	1,7984E+03	4,3160E+04	1,1204E+08	1,1000E+05	1,2165E+04	1,6911E+07	2,5929E+00	2,1514E+01
	19	2,5946E+03	4,9107E+04	1,9346E+08	1,3941E+05	1,7247E+04	5,2512E+07	3,9891E+00	2,1555E+01
	25	4,3199E+03	7,9515E+04	3,0558E+08	1,5725E+05	2,0967E+04	1,6984E+08	6,9864E+00	2,1610E+01
	Promedio	1,8326E+03	4,3331E+04	1,1779E+08	1,1306E+05	1,2851E+04	3,2740E+07	2,6772E+00	2,1519E+01
	Desvio	1,1740E+03	1,3484E+04	9,0900E+07	2,8974E+04	4,7228E+03	4,2925E+07	1,9705E+00	5,4152E-02

Tabla 21: Funciones de la 01 a la 08 dPSO en 10 dimensiones

FES	Problema	09	10	11	12	13	14	15	16
1,00E+03	1	9,4075E+02	1,5980E+03	9,0748E+01	1,0872E+07	1,6956E+05	2,3834E+01	1,2964E+03	1,3751E+03
	7	1,1848E+03	2,0470E+03	9,8717E+01	1,5159E+07	2,3425E+07	2,4773E+01	1,6386E+03	1,7241E+03
	13	1,3047E+03	2,3616E+03	9,9383E+01	1,8240E+07	3,0366E+07	2,4910E+01	1,6866E+03	1,8299E+03
	19	1,4052E+03	2,4935E+03	1,0305E+02	2,0223E+07	3,6602E+07	2,5073E+01	1,8767E+03	1,9292E+03
	25	1,7531E+03	2,8019E+03	1,0768E+02	2,4369E+07	4,7848E+07	2,5270E+01	2,2016E+03	2,1093E+03
	Promedio	1,3009E+03	2,2831E+03	1,0067E+02	1,7591E+07	2,8850E+07	2,4861E+01	1,7463E+03	1,8351E+03
	Desvio	1,8221E+02	2,8842E+02	4,2492E+00	3,2999E+06	1,2378E+07	3,4938E-01	1,9678E+02	1,6911E+02
1,00E+04	1	1,0072E+03	1,6035E+03	8,8137E+01	1,0072E+07	3,5233E+04	2,4267E+01	1,4087E+03	1,3218E+03
	7	1,1380E+03	1,8798E+03	9,1012E+01	1,5502E+07	3,4907E+06	2,4890E+01	1,5907E+03	1,5918E+03
	13	1,1990E+03	2,1334E+03	9,3331E+01	1,8229E+07	1,0698E+07	2,5033E+01	1,7271E+03	1,7488E+03
	19	1,2878E+03	2,3715E+03	9,5368E+01	2,0197E+07	1,8006E+07	2,5187E+01	1,8260E+03	1,8191E+03
	25	1,5103E+03	2,7836E+03	1,0543E+02	2,4069E+07	2,9708E+07	2,5697E+01	2,2302E+03	2,1208E+03
	Promedio	1,2284E+03	2,1500E+03	9,3861E+01	1,7959E+07	1,1451E+07	2,5036E+01	1,7114E+03	1,7320E+03
	Desvio	1,2639E+02	3,2744E+02	4,5182E+00	3,3524E+06	8,6746E+06	3,1956E-01	1,9668E+02	1,6580E+02
1,00E+05	1	8,3928E+02	1,4953E+03	9,0953E+01	1,0092E+07	1,1786E+06	2,4425E+01	1,1440E+03	9,6185E+02
	7	9,8836E+02	1,7076E+03	9,7519E+01	1,4275E+07	3,2696E+06	2,4919E+01	1,4093E+03	1,2687E+03
	13	1,1109E+03	1,8334E+03	1,0046E+02	1,5924E+07	7,8209E+06	2,5024E+01	1,5500E+03	1,4497E+03
	19	1,2361E+03	2,0461E+03	1,0186E+02	1,7779E+07	1,3502E+07	2,5197E+01	1,6103E+03	1,5770E+03
	25	1,3349E+03	2,3318E+03	1,0963E+02	2,3051E+07	3,1335E+07	2,5587E+01	1,7303E+03	2,0052E+03
	Promedio	1,1111E+03	1,8710E+03	1,0002E+02	1,6191E+07	9,6014E+06	2,5063E+01	1,4963E+03	1,4184E+03
	Desvio	1,3905E+02	2,3151E+02	4,3020E+00	3,4328E+06	8,0120E+06	2,9639E-01	1,5265E+02	2,3643E+02
5,00E+05	1	1,5223E+02	2,8754E+02	6,4482E+01	2,4879E+04	3,8642E+01	2,3962E+01	3,5796E+02	1,3976E+02
	7	2,3260E+02	3,8206E+02	7,4789E+01	4,0140E+04	3,6967E+03	2,4263E+01	4,4175E+02	3,3591E+02
	13	2,6368E+02	5,2580E+02	8,7538E+01	7,2906E+04	1,1972E+06	2,4418E+01	1,1849E+03	4,0088E+02
	19	2,8078E+02	1,2402E+03	9,0160E+01	1,1905E+05	2,0569E+06	2,4455E+01	1,2764E+03	8,8562E+02
	25	9,7168E+02	1,5187E+03	9,2071E+01	6,7723E+06	5,3657E+06	2,4620E+01	1,4985E+03	1,3381E+03
	Promedio	3,6202E+02	7,0880E+02	8,3035E+01	3,4834E+05	1,3437E+06	2,4357E+01	9,2339E+02	6,1000E+02
	Desvio	2,5941E+02	4,4301E+02	9,1131E+00	1,3392E+06	1,4569E+06	1,5684E-01	4,1748E+02	3,7526E+02

Tabla 22: Funciones de la 09 a la 16 dPSO en 10 dimensiones

FES	Problema	17	18	19	20	21	22	23	24	25
1,00E+03	1	1,7062E+03	1,4898E+03	1,1286E+03	1,4761E+03	1,7218E+03	1,9896E+03	1,6523E+03	1,6670E+03	1,7787E+03
	7	2,0394E+03	1,6448E+03	1,6607E+03	1,6907E+03	1,8239E+03	2,5807E+03	1,8467E+03	1,8980E+03	2,0507E+03
	13	2,2921E+03	1,7192E+03	1,7576E+03	1,7544E+03	1,8758E+03	3,4607E+03	1,9341E+03	1,9583E+03	2,1204E+03
	19	2,5804E+03	1,8030E+03	1,8078E+03	1,8592E+03	1,9406E+03	6,2111E+03	1,9869E+03	2,0679E+03	2,2703E+03
	25	3,3350E+03	2,1177E+03	1,9832E+03	2,0667E+03	2,3015E+03	3,5391E+04	2,0992E+03	2,5025E+03	1,4763E+05
	Promedio	2,3194E+03	1,7278E+03	1,7273E+03	1,7714E+03	1,9092E+03	5,9144E+03	1,9129E+03	2,0086E+03	1,0382E+04
	Desvio	3,6864E+02	1,3198E+02	1,7189E+02	1,3808E+02	1,3674E+02	6,9946E+03	1,0831E+02	2,0267E+02	2,9527E+04
1,00E+04	1	1,4264E+03	1,5204E+03	1,5118E+03	1,4528E+03	1,5832E+03	1,9490E+03	1,6254E+03	1,6982E+03	1,7572E+03
	7	1,7827E+03	1,5908E+03	1,6556E+03	1,6079E+03	1,7175E+03	2,3760E+03	1,6958E+03	1,8748E+03	1,8661E+03
	13	2,1671E+03	1,6644E+03	1,7231E+03	1,6849E+03	1,7900E+03	2,7349E+03	1,7482E+03	1,9455E+03	1,9972E+03
	19	2,3626E+03	1,7161E+03	1,7757E+03	1,7783E+03	1,8723E+03	3,4864E+03	1,8339E+03	2,0723E+03	2,6278E+03
	25	2,8625E+03	1,8768E+03	2,0893E+03	2,0153E+03	2,1094E+03	1,0828E+05	2,0900E+03	2,4661E+03	Nan
	Promedio	2,1146E+03	1,6686E+03	1,7274E+03	1,7021E+03	1,8038E+03	7,7479E+03	1,8022E+03	1,9852E+03	2,0842E+03
	Desvio	3,9421E+02	9,8244E+01	1,2878E+02	1,5170E+02	1,1141E+02	2,1152E+04	1,5250E+02	1,8203E+02	5,0196E+02
1,00E+05	1	1,1196E+03	1,3949E+03	1,3704E+03	1,4591E+03	1,4865E+03	1,7397E+03	1,5333E+03	1,6645E+03	1,8043E+03
	7	1,6205E+03	1,5168E+03	1,5416E+03	1,5234E+03	1,6099E+03	2,0355E+03	1,6120E+03	1,7378E+03	1,9484E+03
	13	1,7784E+03	1,6038E+03	1,6140E+03	1,5877E+03	1,6638E+03	2,1614E+03	1,6644E+03	1,8229E+03	2,0948E+03
	19	1,9262E+03	1,6527E+03	1,7459E+03	1,6677E+03	1,7041E+03	2,3684E+03	1,7155E+03	1,8887E+03	Nan
	25	2,5964E+03	1,8140E+03	1,8423E+03	1,9234E+03	1,8478E+03	6,5760E+03	1,7898E+03	2,4233E+03	Nan
	Promedio	1,7848E+03	1,5951E+03	1,6254E+03	1,6159E+03	1,6621E+03	2,4416E+03	1,6645E+03	1,8464E+03	1,9873E+03
	Desvio	3,4722E+02	1,0429E+02	1,3384E+02	1,2635E+02	9,0134E+01	9,4860E+02	7,3580E+01	1,5736E+02	1,2941E+02
5,00E+05	1	6,3607E+02	9,6885E+02	9,4653E+02	9,4918E+02	5,0000E+02	1,0312E+03	8,1453E+02	2,0000E+02	1,6725E+03
	7	7,1198E+02	1,0364E+03	1,0439E+03	1,0290E+03	5,0000E+02	1,1416E+03	1,0614E+03	2,0000E+02	1,7710E+03
	13	1,0292E+03	1,4186E+03	1,3833E+03	1,3588E+03	1,0609E+03	1,2017E+03	1,1201E+03	1,1036E+03	1,8238E+03
	19	1,2202E+03	1,4383E+03	1,4474E+03	1,4468E+03	1,1696E+03	1,5388E+03	1,4530E+03	1,4124E+03	Nan
	25	1,6256E+03	1,5100E+03	1,5055E+03	1,5975E+03	1,5764E+03	2,1540E+03	1,6157E+03	1,7332E+03	Nan
	Promedio	1,0213E+03	1,2994E+03	1,2467E+03	1,2656E+03	9,7162E+02	1,3437E+03	1,2147E+03	8,4829E+02	1,8093E+03
	Desvio	2,9108E+02	2,0498E+02	2,1817E+02	2,2438E+02	3,7553E+02	3,1881E+02	2,4830E+02	6,5117E+02	1,3024E+02

Tabla 23: Funciones de la 17 a la 25 dPSO en 10 dimensiones

Problema	1	7	13	19	25	Promedio	Desvio	Exito
01	-	-	-	-	-	-	-	0%
02	-	-	-	-	-	-	-	0%
03	-	-	-	-	-	-	-	0%
04	-	-	-	-	-	-	-	0%
05	-	-	-	-	-	-	-	0%
06	-	-	-	-	-	-	-	0%
07	-	-	-	-	-	-	-	0%
08	-	-	-	-	-	-	-	0%
09	-	-	-	-	-	-	-	0%
10	-	-	-	-	-	-	-	0%
11	-	-	-	-	-	-	-	0%
12	-	-	-	-	-	-	-	0%
13	-	-	-	-	-	-	-	0%
14	-	-	-	-	-	-	-	0%
15	-	-	-	-	-	-	-	0%
16	-	-	-	-	-	-	-	0%
17	-	-	-	-	-	-	-	0%
18	-	-	-	-	-	-	-	0%
19	-	-	-	-	-	-	-	0%
20	-	-	-	-	-	-	-	0%
21	-	-	-	-	-	-	-	0%
22	-	-	-	-	-	-	-	0%
23	-	-	-	-	-	-	-	0%
24	-	-	-	-	-	-	-	0%
25	-	-	-	-	-	-	-	0%

Tabla 24: Funciones de la 01 a la 25 dPSO en 10 dimensiones

2. Aspectos de Implementación

En este apartado se especifica la forma en qué intercambiaran información el sistema de Gestión de Cabina existente y el módulo de optimización basado en algoritmos evolutivos.

Especificación de Interfaces

A continuación se detallan los requerimientos a cumplir:

1. La interface será un archivo de texto.
2. Se generará un ID del lado del sistema de cabina.
3. El ID será parte del nombre de los archivos de entrada y salida del algoritmo.
4. Los archivos de texto estarán bloqueados durante su proceso de lectura/escritura, para asegurar la consistencia.
5. Los archivos serán del tipo texto.
6. Los valores de las distintas columnas estarán separados por un espacio.
7. Las columnas numéricas deberán respetar el ancho, y los valores deberán estar formateados.
8. Para las columnas de texto, el ancho es sólo un valor máximo.
9. Las columnas de texto no se completarán con espacios, ni podrán tener espacios intermedios.
10. Existirá un archivo de texto de control donde se detallará la información de secuencia y demás parámetros generales.
11. El algoritmo de asignación automática deberá poder ejecutarse desde línea de comando.

Especificación de archivos

El sistema de Administración de Cabina generará tres archivos, el primero con las prestaciones, el segundo con los móviles disponibles y el tercero será un archivo de control. El primero de ellos se llamara **prestnnnnnn**, siendo *nnnnnn* un número único de 6 dígitos (secuencia).

El segundo de ellos se llamara **movilnnnnnn**, siendo *nnnnnn* un número único de 6 dígitos (secuencia). Ambos números deberán coincidir.

En el archivo control, se encontrará el valor de la secuencia que determina los nombres de los archivos anteriores, así como otros valores de parámetros de funcionamiento.

El módulo de Asignación devolverá un archivo llamado **asignnnnnn**, con la misma lógica que los anteriores. Este archivo devolverá la asignación automática realizada por el algoritmo.

Todos los archivos descritos anteriormente serán del tipo texto con formato de registro de longitud fija, y tendrán extensión **.txt**.

En el siguiente párrafo se detalla el formato de cada uno de estos archivos, así como el detalle del contenido de cada uno de ellos.

El algoritmo de asignación automática buscará los archivos de entrada en el directorio donde se ejecuta, y dejará el archivo de salida en el mismo directorio. El programa de asignación no borrará los archivos de entrada. Asimismo, en caso de existir el de salida, el mismo será sobrescrito.

Detalle de Archivos

En este párrafo se detallan todos los archivos de interface. A continuación se detallan las convenciones generales

1. Los archivos son de longitud variable, tipo texto, usando el espacio como separador de columnas.
2. Los números deben formatearse de acuerdo al detalle.
3. Las columnas textos no podrán tener espacios intermedios.
4. No deberán utilizarse caracteres especiales.
5. Las columnas tipo texto se alinean a izquierda, y no se completan con espacios.
6. Las columnas tipo número se alinean a derecha, y se completan con ceros.
7. Los tiempos estarán expresados en segundos, y serán relativos a la ejecución del algoritmo. Por ejemplo, si un móvil comienza a operar 1 hora y 10 minutos luego de la generación del archivo, en la columna HoraDesde deberá figurar el número 4200.
8. Las velocidades promedio será del tipo lineal, expresada km/h, y enteros.

Prestaciones

En este archivo se detallarán todas las prestaciones pendientes de asignar. A continuación se detalla la estructura del archivo.

El archivo tendrá tantas líneas como prestaciones pendientes se encuentren (ver tabla 25).

Columna	Contenido	Tipo	Exten-sión	Observaciones
Código	Código de prestación	Texto	10	
Longitud	Coordenada de la prestación	Numérico	9	Sin separador de coma
Latitud	Coordenada de la prestación	Numérico	9	Sin separador de coma
Urgente	Priorizar prestación	Texto	1	S/N
Hora Llamado	Tiempo transcurrido desde llamado	Numérico	7	En segundos
Reclamos	Indica si tiene reclamos	Texto	1	S/N
Tipo de socio	Indica si es socio Directo, Cápita o Tercero	Texto	1	D/C/T
Pediátrico	Indica si es paciente es Pediátrico, Mayor u Otro	Texto	1	P/M/O
Despacho	Código de despacho por defecto	Texto	10	
Tiempo Promedio	Tiempo promedio de atención	Numérico	5	En segundos
Desvio Promedio	Desvivo medio en tiempos de atención	Numérico	5	En segundos
Tipo Prestación	Categoría de la Prestación (Roja/Amarilla/Verde)	Texto	1	R/A/V
Zona	Zona de la prestación (Localidad)	Texto	n	

Tabla 25: Archivos de prestaciones a resolver

El nombre de este archivo será **prestnnnnnn.txt**, siendo **nnnnnn** un número único de 6 dígitos (secuencia) que se encuentra en el archivo de control (**control.txt**).

Móviles

En este archivo se detallan los móviles disponibles al momento de ejecución del proceso, así como los móviles disponibles en el futuro (ver tabla 26).

Tener en consideración que de acuerdo a las prestaciones pendientes, se deben incorporar las guardias a futuro, a los efectos de asignar todas las prestaciones

Columna	Contenido	Tipo	Exten-sión	Observaciones
Código	Código de móvil	Texto	10	
Longitud	Coordenada de la prestación. Última ubicación	Numérico	9	Sin separador de coma
Latitud	Coordenada de la prestación. Última ubicación	Numérico	9	Sin separador de coma
Hora Desde	Hora de comienzo de disponibilidad. Comienzo de guardia, o tiempo aproximado para liberarse de las prestaciones asignadas	Numérico	7	En segundos, tomando como base la fecha de generación del archivo. 0 = disponible inmediatamente
Hora Hasta	Hora de fin de disponibilidad. Fin de la guardia	Numérico	7	En segundos, tomando como base la fecha de generación del archivo
Pediátrico	Indica si el móvil es pediátrico exclusivo	Texto	1	S/N
Hace Pediatrico	Indica si el móvil atiende pediátrico	Texto	1	S/N
Hace Mayor	Indica si el móvil atiende mayores	Texto	1	S/N
Tercero	Indica si el móvil es de tercero	Texto	1	S/N
Tipo Facturación	Indica si el móvil tiene costo fijo o variable	Texto	1	F/V
Tipo de móvil	Típo de móvil (UTIM, Auto, Camioneta)	Texto	1	U/A/C
Velocidad	Velocidad promedio lineal, en km/h	Numérico	3	Números enteros

Desviopromedio	Desvio promedio en velocidad lineal	Numérico	3	Números enteros
Despacho	Código de despacho por defecto	Texto	10	
Costo de Operación	Costo de uso del móvil, en categoría Alto, Mediano o Bajo	Texto	1	A/M/B
Radio	Radio de acción	Numérico		0 si no tiene radio
Longitud Acción	Coordenada del radio de acción	Numérico	9	Sin separador de coma
Latitud Acción	Coordenada del radio de acción	Numérico	9	Sin separador de coma
Zona	Zona de acción (Localidades)	Texto		Texto sin espacios. Lista de localidades separadas con &. Se utiliza para buscar la zona de prestación dentro de este texto. Sino tiene zonas, poner "T"

Tabla 26: Archivo con atributos de los móviles

El nombre de este archivo será **movilnnnnnn.txt**, siendo **nnnnnn** un número único de 6 dígitos (secuencia) que se encontrará definido en el archivo de control (**control.txt**)

Control

Este archivo controlará la ejecución del proceso, así como los parámetros generales de funcionamiento. Este archivo tendrá solo una línea (ver tabla 27).

Columna	Contenido	Tipo	Extensión	Observaciones
Secuencia	Secuencia de ejecución	Numérico	6	Secuencia de ejecución del proceso
Número de Prestaciones	Cantidad de prestaciones	Numérico	6	Número entero. Debe coincidir con las líneas del archivo
Número de móviles	Cantidad de móviles Numérico	6		Número entero. Debe coincidir con la cantidad de móviles.
Fecha Proceso	Fecha de ejecución. Auditoria	Texto	12	DDMMYYYYH Hmm

Tabla 27: Archivo de control del proceso

Este archivo se llamará **control.txt**. El mismo se mantendrá bloqueado durante la ejecución del algoritmo.

Asignación

Este archivo será generado por el proceso de asignación automática (ver tabla 28).

Columna	Contenido	Tipo	Extensión	Observaciones
CódigoPrestación	Código de prestación	Texto	10	
CódigoMóvil	Código Móvil	Texto	10	
HoraPrestación	Hora de Prestación. Define orden y horario estimado	Numérico	7	En segundos
Demora Total	Demora total en segundos, desde llamado	Numérico	7	En segundos
Distancia	Distancia en km desde la prestación anterior	Numérico		
Distancia Origen	Distancia en km desde lugar origen	Numérico		
Detalle	Especifica detalle de asignación	Texto	100	

Tabla 28: Archivo de asignación

En caso que no se pueda encontrar ningún móvil para atender la prestación, la columna de **CodigoMovil** y **HoraPrestación** pueden estar vacías.

En la columna de detalle se define cualquier texto de interés en la asignación realizada.

Este archivo se llamará **asignnnnnn.txt**, siendo **nnnnnn** un número único de 6 dígitos (secuencia) que se encontrará definido en el archivo de control (**control.txt**).

Índice de figuras

Lista de algoritmos

Bibliografía

[1] Dantzig George B. Linear Programming and Extensions. Princeton University Press, 1963.

[2] Magnanti Thomas L. Twenty years of mathematical programming. Contributions to Operations Research and Economics: the twentieth anniversary of CORE, pages 163–227, 1989.

[3] Goldfarb D. and Todd M. Linear programming. Handbooks in operations research and management science, 1:73 – 170, 1989.

[4] Dantzig George B. Linear Programming and Extensions. Princeton University Press, 1963.

[5] Von Neumann J. Morgenstern. Theory of Games and Economic Behavior. Princeton University Press, Princeton, 1944.

[6] Gill P., Murray W., Saunders M., and Wright M. Constrained nonlinear programming. Stanford University, Dept. of Operations Research, Systems Optimization Laboratory, 1987.

[7] Nemhauser G. and Wolsey L. Integer and combinatorial optimization. New York Wiley, 1988.

[8] Weise Thomas. Global optimization algorithms. theory and application (ebook). Website, 2012.

http://www.it-weise.de/projects/book.pdf.

[9] Robert Fourer. Formulating an optimization model: An introductory example. Website.

http://www.iems.northwestern.edu/~4er/.

[10] Leguizamon Guillermo. Metaheur´ısticas para Problemas de Optimizaci´on Restringidos. PhD thesis, Facultad de Ciencias F´ısico Matem´aticas y Naturales. UNSL, 2004.

[11] Paterson Norman. Genetic porgramming with context-sensitive grammars. PhD thesis, Saint Andrew's University, 2002.

[12] Radcliffe Nicholas and Surry Patrick. Fundamental limitations on search algorithms: Evolutionary computing in perspective. Lecture Notes in Computer Science, pages 275–291, 1995.

[13] Zäpfel G., Braune R., and B¨ogl M. Metaheuristic Search Concepts A Tutorial with Applications to Production and Logistics. 2010.

[14] Papalambros P. and Wilde D. Principles of Optimal Design: Modeling and Computation.

Cambridge University Press, New York, 1988.

[15] Rice John. Mathematical Statistics and Data Analysis. Duxbury Press, 1999.

[16] Nemhauser G.L., Rinnooy Kan A.H.G., and Todd M.J. Handbooks in Operations Research and

Management Science. North-Holland/Elsevier, 1989.

[17] Bader David, HartWilliam, and Phillips Cynthia. Parallel algorithm design for branch and bound.

Tutorials on Emerging Methodologies and Applications in Operations Research, pages 1–44, 2004.

[18] Casado L. G. and Garc´ıa I. Work load balance approaches for branch and bound algorithms on distributed systems. 7th Euromicro Workshop on Parallel and Distributed Processing, pages 155–162, 1999.

[19] Casado L. G., Garc´ıa I., and Csendes T. A new multisection technique in interval methods for global optimization. Computing, 65:263–269, 2000.

[20] Glover F. and Kochenberger G. Handbook of Metaheuristics. Kluwer Academic Publishers, 2003.

[21] Fraser AS. Monte carlo analyses of genetic models. Nature, 181, 1958.

[22] Holland John H. Adaptation in Natural and Artificial Systems. MIT Press, 1992.

[23] Koza John R. Genetic Programming. MIT Press, 1992.

[24] Beni G. and Wang J. Swarm intelligence in cellular robotic systems. NATO Advanced Workshop on Robots and Biological Systems, pages 26–30, 1989.

[25] Darwin Charles. On the Origin of Species. Bantam Dell, 1859.

[26] Coello Coello Carlos. Constraint-handling techniques used with evolutionary algorithms. In GECCO (Companion), pages 1137–1160, 2011.

[27] Zakian Vladimir. New formulation for the method of inequalities. In Proceedings of the Institution of Electrial Engineers, volume 126, pages 579–584, 1979.

[28] Zakian Vladimir. New formulation for the method of inequalities. Systems and Control Encyclopedia, 5:3206–3215, 1987.

[29] Goldberg David E. The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Addison-Wesley, 2002.

[30] Whitley D. A genetic algorithm tutorial. Statistics and Computing, 4:65–85, 1994.

[31] Goldberg David E. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.

[32] Miller Brad L. and Goldberg David E. Genetic algorithms, tournament selection, and the e_ects of noise. Complex Systems, 9:193–212, 1995.

[33] Byoung tak Zhang and Jung jib Kim. Comparison of selection methods for evolutionary optimization. Evolutionary Optimization, 2:55–70, 2000.

[34] Baker James E. Adaptive selection methods for genetic algorithms. In Proceedings of the 1st International Conference on Genetic Algorithms, pages 101–111, Hillsdale, NJ, USA, 1985. L. Erlbaum Associates Inc.

[35] Hancock Peter J. B. An empirical comparison of selection methods in evolutionary algorithms.

[36] Eshelman Larry J. The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In Proceedings of the First Workshop on Foundations of Genetic Algorithms, pages 265–283. Morgan Kaufmann, 1991.

[37] Gwiazda Tomasz D. Genetic Algorithms Reference Vol.1 Crossover for single-objective numerical optimization problems. -omianki : TOMASZGWIAZDA E-BOOKS, cop, 2006.

[38] Chawdhry P.K. Soft computing in engineering design and manufacturing. London: Springer, 1998.

[39] Ahmed Zakir H. Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. International Journal of Biometric and Bioinformatics, 3, 2010.

[40] Herrera F., Lozano M., and S´anchez A.M. A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study. International Journal of Intelligent Systems, 18:309–338, 2003.

[41] Eiben A. Multiparent recombination operators in evolution strategies. Evolutionary Computation, 5:347–365, 1997.

[42] Eshelman L. J. and Scha_er J. D. Real- Coded Genetic Algorithms and Interval Schemata, Foundations of Genetic Algorithms. 2, Morgan Kaufman Publishers, San Mateo, 1993.

[43] Shigeyoshi Tsutsui, Masayuki Yamamura, and Takahide Higuchi. Multi-parent recombination with simplex crossover in real coded genetic algorithms. In In Proc. of the Genetic and Evolutionary Computation Conference, volume 1, pages 657–664, 1999.

[44] Ochoa G. Setting the mutation rate: Scope and limitations of the 1/l heuristic. In Proceedings of the Genetic and Evolutionary Computation Conference, pages 495–502. Morgan Kaufmann Publishers, 2002.

[45] Andre J., Siarry P., and Dognon T. An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization. Advances in engineering software, 32:49–60, 2001.

[46] Srinivas M. and Patnaik L. Genetic algorithms. a survey. IEEE Computer Magazine, pages 17–26, 1994.

[47] Beasley D., Bull D., and Martin R. An overview of genetic algorithms: Part 1, fundamentals.

Technical report. http://home.ifi.uio.no/jimtoer/GAOverview1.pdf.

[48] Fogel D. B. An introduction to simulated evolutionary optimization. IEEE Transactions on Neural Networks, 5:3–14, 1994.

[49] Smith J. E. and Fogarty T. C. Operator and parameter adaptation in genetic algorithms. Soft computing : a fusion of foundations, methodologies and applications, 92:81–87, 1997.

[50] Nichael Lynn Cramer. A representation for the adaptive generation of simple sequential programs. In Proceedings of an International Conference on Genetic ALgorithms and their Applications, pages 24–26, 1985.

[51] Koza J.R. Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems. Technical Report STAN-CS-90-1314, Stanford University Computer Science Department, 1990.

[52] Koza J.R. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, 1992.

[53] Koza J.R. Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, 1994.

[54] Koza J.R., Bennett F.H., D. Andre, and Keane M.A. Genetic Programming III: Darwinian Invention and Problem Solving. Morgan Kaufmann, 1999.

[55] Koza J.R., M.A. Keane, M.J. Streeter, Mydlowec W., Yu J., and G. Lanza. Genetic Programming

IV: Routine Human-Competitive Machine Intelligence. Kluwer Academic Publishers, 2003.

[56] Banzhaf W., Nordin P., Keller R.E., and Francone F.D. Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications. Morgan Kaufmann, 1998.

[57] Korns M. Large-scale, time-constrained symbolic regression-classification. In Rick L. Riolo, Terence Soule, and Bill Worzel, editors, Genetic Programming Theory and Practice V, Genetic and Evolutionary Computation, chapter 4, pages 53–68. Springer, 2007.

[58] Korns M. Symbolic regression of conditional target expressions. In Rick Riolo, Una-May O'Reilly, and Trent McConaghy, editors, Genetic Programming Theory and Practice VII, Genetic and Evolutionary Computation, pages 211–228. Springer US, 2010.

[59] Korns M. Abstract expression grammar symbolic regression. In Rick Riolo, Trent McConaghy, and Ekaterina Vladislavleva, editors, Genetic Programming Theory and Practice VIII, volume 8 of Genetic and Evolutionary Computation, pages 109–128. Springer New York, 2011.

[60] Beyer Hans Georg and Schwefel Hans Paul. Evolution strategies. a comprehensive introduction. 1(1):3–52, 2002.

[61] Schwefel H. P. Evolution strategies: A family of non-linear optimization techniques based on imitating some principles of organic evolution. Annals of Operations Research, 1:165–167, 1984.

[62] Hansen Nikolaus and Kern Stefan. Evaluating the cma evolution strategy on multimodal test functions. In Parallel Problem Solving from Nature - PPSN VIII, volume 3242 of Lecture Notes in Computer Science, pages 282–291. Springer Berlin Heidelberg, 2004.

[63] Storn Rainer and Price Kenneth. Di_erential evolution. a simple and e_cient heuristic for global optimization over continuous spaces. J. of Global Optimization, 11(4):341–359, December 1997.

[64] Rainer Storn. On the usage of di_erential evolution for function optimization. In in NAFIPS'96, pages 519–523. IEEE, 1996.

[65] Feoktistov Vitaliy. Di_erential Evolution: In Search of Solutions (Springer Optimization and Its Applications). Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[66] Chakraborty Uday K. Advances in Di_erential Evolution. Springer Publishing Company, Incorporated, 1 edition, 2008.

[67] Professor Kaelo. Some population-set based methods for unconstrained global optimization.

PhD thesis, School of Computational and Applied Mathematics, Witwatersrand University, Johannesburg, South Africa, 2005.

[68] Professor Kaelo and Montaz Ali M. A numerical study of some modified di_erential evolution algorithms. European Journal of Operational Research, 169:1176–1184, 2006.

[69] Professor Kaelo and Montaz Ali M. Di_erential evolution algorithms using hybrid mutation. Computational Optimization and Applications, 37:231–246, 2007.

[70] Liu J. and Lampinen J. A fuzzy adaptive di_erential evolution algorithm. Soft Computing, 9:448–462, 2005.

[71] Qin A.K. and Suganthan P.N. Self-adaptive di_erential evolution algorithm for numerical optimization. In Proceedings of the IEEE congress on evolutionary computation (CEC), pages 1785–1791, 2005.

[72] Beni G. andWang J. Swarm intelligence in cellular robotic systems. In Proceed. NATO Advanced Workshop on Robots and Biological Systems, pages 26–30, 1989.

[73] D. Karaboga. Artificial bee colony algorithm. Scholarpedia, 5(3):6915, 2010.

[74] Kaveh A. and Talatahari S. A novel heuristic optimization method: charged system search. Acta Mechanica, 213:267–289, 2010.

[75] Yang X. and Deb S. Cuckoo search via levy flights. In IEEE Publications, editor, Proc. of World Congress on Nature and Biologically Inspired Computing NaBIC 2009, pages 210–214, 2009.

[76] Farmer J. D., Packard N. H., and Perelson A. S. The immune system, adaptation, and machine learning. Phys. D, 2(1-3):187–204, October 1986.

[77] Bersini Hugues and Varela Francisco. Hints for adaptive problem solving gleaned from immune

networks. In Parallel Problem Solving from Nature, volume 496 of Lecture Notes in Computer Science, pages 343–354. Springer Berlin / Heidelberg, 1991.

[78] Dasgupta D., editor. Artificial Immune Systems and Their Applications. Springer-Verlag, Inc. Berlin, 1999.

[79] Dorigo M. Optimization, Learning and Natural Algorithms. PhD thesis, Politecnico di Milano, Italie, 1992.

[80] Colorni A., Dorigo M., and Maniezzo V. Distributed optimization by ant colonies. In Actes de la premi`ere conf´erence europ´eenne sur la vie artificielle, pages 134–142, Paris, France, 1991. Elsevier Publishing.

[81] Dorigo M. and St¨utzle T. Ant Colony Optimization. Mit Press, 2004.

[82] Leguizamon G. and Michalewicz Z. A new version of ant system for subset problems. In Proceedings of the 1999 Congress on Evolutionary Computation(CEC 99), volume 2, pages 1458–1464, 1999.

[83] Leguizam´on G., Michalewicz Z., and Sch¨utz M. An ant system for the maximum independent set problem. In In Proceedings of VII Argentine Congress of Computer Science (CACIC 2001), pages 1027–1040, 2001.

[84] Kennedy J. and Eberhart R. Particle swarm optimization. IEEE International Conference on Neural Networks, IV:1942–1948, 1995.

[85] Kennedy J. and Eberhart R. A discrete binary version of the particle swarm algorithm. World Multiconference on Systemics, Cybernetics and Informatics (WMSCI), pages 4104–4109, 1997.

[86] Russell Stuart J. and Norvig Peter. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice-Hall International Edition, 2003.

[87] Kirkpatrick Scott. Optimization by simulated annealing: Quantitative studies. Journal of Statistical Physics, 34:975–986, 1984.

[88] Cerny V. Thermodynamical approach to the traveling salesman problem: An e_cient simulation algorithm. Journal of Optimization Theory and Applications, 45:41–51, 1985.

[89] Granville V., Krivanek M., and Rasson J. P. Simulated annealing: A proof of convergence. 1994.

[90] Glover Fred and McMillan Claude. The general employee scheduling problem: an integration of ms and ai. Computers and Operations Research, 13(5):563–573, May 1986.

[91] Glover Fred. Tabu search - part 1. 1989.

[92] Glover Fred. Tabu search - part 2. 1990.

[93] Glover Fred. Tabu search: A tutorial. Interfaces, 20:74–94, 1990.

[94] Sean Luke. Essentials of Metaheuristics. Lulu, 2009.

[95] Scha_er David, Eshelman Larry, and O_utt Daniel. Spurious correlations and premature convergence in genetic algorithms. In In Proceedings of the First Workshop on Foundations of Genetic Algorithms (FOGA).

[96] Toscano Pulido G. and Coello Coello Carlos. The micro genetic algorithm 2: Towards online adaptation in evolutionary multiobjective optimization. In EMO, pages 252–266, 2003.

[97] L´opez J., Lanzarini L., and De Giusti A. Varmopso: Multi- objective particle swarm optimization with variable population size. In Advances in Artificial Intelligence IBERAMIA 2010, 2010.

[98] Liepins Gunar E. and Vose Michael D. Deceptiveness and genetic algorithm dy- namics. In In Proceedings of the First Workshop on Foundations of Genetic Algorithms (FOGA).

[99] Bergman Aviv and Feldman Marcus W. Recombination dynamics and the fitness landscape. In Physica D: Nonlinear Phenomena.

[100] Kolarov Krasimir. Landscape ruggedness in evolutionary algorithms. In In Proceedings of the IEEE Conference on Evolutionary Computation.

[101] Stagge Peter and Igel Christian. Structure optimization and isomorphisms. In In Theoretical Aspects of Evolutionary Computing.

[102] Reidys Christian M. and Stadler Peter. Neutrality in fitness landscapes. In Applied Mathematics and Computation.

[103] Shackleton Mark, Shipman Rob, and Ebner Marc. An investigation of redundant genotypephenotype mappings and their role in evolutionary search. In In Proceedings of the 2000 Congress on Evolutionary Computation CEC00.

[104] Alba Enrique. Parallel Metaheuristics: A New Class of Algorithms. John Wiley and Sons, 2005.

[105] Luque G. and Alba E. Parallel Genetic Algorithms. Theory and Real World Applications. Springer-Verlag, 2011.

[106] Alba E., Blum C., Isasi P., and Le´on C. G´omez J.A. Optimization Techniques for Solving Complex Problems. Wiley, 2009.

[107] Alba E. and Dorronsoro B. Cellular Genetic Algorithms. Springer-Verlag, 2008.

[108] Alba E. and Dorronsoro B. Computing nine new best-so-far solutions for capacitated vrp with a cellular ga. Information Processing Letters, 98:225–230, 2006.

[109] Giacobini M., Tomassini M., Tettamanzi A., and Alba E. The selection intensity in cellular evolutionary algorithms for regular lattices. IEEE Transactions on Evolutionary Computation, 9:489–505, 2005.

[110] Alba E. and Dorronsoro B. The exploration/exploitation tradeo_ in dynamic cellular genetic algorithms. IEEE Transactions on Evolutionary Computation, 9:126–142, 2005.

[111] Nedjah N., Alba E., and de Macedo Mourelle L. Parallel Evolutionary Computations. Springer-Verlag, 2006.

[112] Nebro A.J., Durillo J.J., Luna F., Dorronsoro B., and Alba E. Mocell: A new cellular genetic algorithm for multiobjective optimization. International Journal of Intelligent Systems, 24:726–746, 2009.

[113] Alba E., Dorronsoro B., Luna F., Nebro A.J., Bouvry P., and Hogie L. A cellular multiobjective genetic algorithm for optimal broadcasting strategy in metropolitan manets. Computer Communications, 30:685–697, 2007.

[114] Plamenka Borovska. E_ciency of parallel metaheuristics for solving combinatorial problems.

In Proceedings of the 2007 international conference on Computer systems and technologies (CompSysTech 07). ACM, New York, NY, USA,, 2007.

[115] Rardin Ronald L. and Uzsoy Reha. Experimental evaluation of heuristic optimization algorithms: A tutorial. Journal of Heuristics, 7(3):261–304, May 2001.

[116] Co_n M.and Saltzman M. J. Statistical analysis of computational tests of algorithms and heuristics. Informs Journal on Computing, 12:24–44, 2000.

[117] Chiarandini Marco, Paquete Lu´ıs, Preuss Mike, and Ridge Enda. Experiments on metaheuristics: Methodological overview and open issues. Technical Report DMF-2007-03-003, The Danish Mathematical Society, Denmark, 2007.

[118] Sheskin David. Handbook of Parametric and Nonparametric Statistical Procedures. Chapman and Hall, 2007.

[119] Harrington David P. and Fleming Thomas R. A class of rank test procedures for censored survival data. Biometrika, 69:553–566, 1982.

[120] Anderson T. W. An Introduction to Multivariate Statistical Analysis. John Wiley & Sons, 2003.

[121] Grunert da Fonseca Viviane, Fonseca Carlos, and Hall Andreia. Inferential performance assessment of stochastic optimisers and the attainment function. In Eckart Zitzler, Lothar Thiele, Kalyanmoy Deb, Carlos Coello Coello, and David Corne, editors, Evolutionary Multi-Criterion Optimization, volume 1993 of Lecture Notes in Computer Science, pages 213–225. Springer Berlin / Heidelberg, 2001.

[122] Bartz-Beielstein Thomas. Experimental Research in Evolutionary Computation - The New Experimentalism. Springer Verlag, 2006.

[123] Liang J. J., Suganthan P. N., and Deb K. Novel composition test functions for numerical global optimization. In Proceedings of IEEE Swarm Intelligence Symposium, pages 68–75, 2005.

[124] Horn Je_rey, Goldberg David E., and Deb Kalyanmoy. Long path problems for mutation-based algorithms, 1992.

[125] ZhongW. C., Liu J., Xue M. Z., and Jiao L. C. A multiagent genetic algorithm for global numerical optimization. IEEE Trans. on Systems, Man and Cybernetics (Part B), 34:1128–1141, 2004.

[126] Deb Kalyanmoy. Multi-objective genetic algorithms: Problem di_culties and construction of test problems. Evolutionary Computation, 7:205–230, 1999.

[127] Coello Coello C. A., Pulido G., and Lechuga M. Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation, 8:256 – 279, 2004.

[128] van den Bergh F. and Engelbrecht A. P. A cooperative approach to particle swarm optimization. IEEE Transactions on Evolutionary Computation, 8:225 – 239, 2004.

[129] Leung Y. W. and Wang Y. P. An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Trans. on Evolutionary Computation, 5:41–53, 2001.

[130] Suganthan P., Hansen N., Liang J., Deb K., Chen Y., Auger A., and Tiwari S. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. In Technical Report, Nanyang Technological University, Singapore, May 2005 AND KanGAL Report 2005005, IIT Kanpur, India.

[131] Liang J. J., Runarsson T. P., Mezura-Montes E., Clerc M., Suganthan1 P., Coello Coello C., and Deb K. Problem definitions and evaluation criteria for the cec 2006. In Special Session on Constrained Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore, 2006.

[132] Mallipeddi R. and Suganthan P. Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization. In Technical Report, Nanyang Technological University, Singapore, 2010.

[133] Huang V., Qin A., Deb K., Zitzler E., Suganthan P., Liang J., Preuss M., and Huband S. Problem definitions for performance assessment of

multi-objective optimization algorithms. In Special Session on Constrained Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore, 2007.

[134] Zitzler E., Deb K., and Thiele L. Comparison of multiobjective evolutionary algorithms on test functions of di_erent di_culty. In Genetic and Evolutionary Computation Conference (GECCO-99): Bird-of-a-feather Workshop on Multi-criterion Optimization Using Evolutionary Methods, July 1999.

[135] Zhang Qingfu, Zhou Aimin, Zhao S., Suganthan P., Liu W., and Tiwari S. Multiobjective optimization test instances for the cec 2009 special session and competition. In Technical Report CES-887, University of Essex and Nanyang Technological University, 2008.

[136] Zitzler Eckart and Laumanns Marco. Test problems and test data for multiobjective optimizers. Website. http://www.tik.ee.ethz.ch/sop/ download/supplementary/testProblemSuite/.

[137] Coello Coello Carlos A. A short tutorial on evolutionary multiobjective optimization, 2001.

[138] K¨oksalan M., Wallenius J., and Zionts S. Multiple Criteria Decision Making: From Early History to the 21st Century. World Scientific, 2011.

[139] Condercet M. Essay on the Application of Analysis to the Probability of Majority Decisions, 1785.

[140] Pareto V. Manual of political economy. Scholars Book Shelf, 1971.

[141] Serafini P. Some considerations about computational complexity for multiobjective combinatorial problems. In J. Jahn andW. Krabs, editors, Recent advances and historical development of vector optimization, volume 294 of Lecture Notes in Economics and Mathematical Systems, Berlin, 1986. Springer-Verlag.

[142] Camerini P., Galbiati G., and Ma_oli F. On the complexity of finding multi-constrained spanning trees. Discrete Applied Mathematics, 5:39–50, 1983.

[143] Deb K. Solving goal programming problems using multi-objective genetic algorithms. In Proceedings of the 1999 Congress on Evolutionary Computation CEC99, pages 77–84, 1998.

[144] Wierzbicki A.P. Reference point approaches. In T. Stewart T. Gal and T. Hanne, editors, Multicriteria Decision Making: Advances in MCDM Models, Algorithms, Theory, and Applications, pages 9.1–9.39, Boston, 1999. Kluwer Academic Publishers.

[145] Reeves G.R. and Gonzalez J.J. A comparison of two interactive mcdm procedures. European Journal of Operational Research, (2):203–209, 1989.

[146] Corner J.L. and Buchanan J.T. Capturing decision maker preference: Experimental comparison of decision analysis and mcdm techniques. European Journal of Operational Research, (1):85–97, 1997.

[147] Deb Kalyanmoy and Chaudhuri Shamik. I-emo: An interactive evolutionary multi-objective optimization tool. In In Pattern Recognition and Machine Intelligence: First International Conference (PReMI-2005), pages 690–695. Springer.

[148] Zujevs A. Model of decision maker as optimization problem for genetic optimization algorithm. In 4-th International conference Applied Information and Communication Technologies, pages 79–86, 2010.

[149] Lotov Alexander V., Bushenkov Vladimir A., and Kamenev Georgy K. Interactive Decision Maps. Approximation and Visualization of Pareto Frontier, volume 89. Springer-Verlag, 2004.

[150] Zitzler E., Laumanns M., and Bleuler S. A tutorial on evolutionary multiobjective optimization.

[151] Peng Cheng. A survey of performance assessment for multiobjective optimizers. In Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on.

[152] Tatsuya Okabe. A critical survey of performance indices for multi-objective optimisation. In Proc. of 2003 Congress on Evolutionary Computation, pages 878–885. IEEE Press, 2003.

[153] Huang V. L., Suganthan P., Qin A., and Baskar S. Multiobjective di_erential evolution with external archive and harmonic distance-based diversity measure.

[154] Deb K., Pratap A., Agarwal S., and Meyarivan T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Transactions on Evolutionary Computation, 6:182–197, 2002.

[155] Zitzler E., Laumanns M., and Thiele L. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K.C. Giannakoglou et al., editors, Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001), pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002.

[156] Nebro, Durillo, García-Nieto, Coello Coello, Luna, and Alba. Smpso: A new pso- based metaheuristic for multi-objective optimization. In IEEE

Symposium on Computational Intelligence in Multicriteria Decision-Making (MCDM 2009), pages 66–73, 2009.

[157] Knowles Joshua and Corne David. Approximating the nondominated front using the pareto archived evolution strategy. Evol. Comput., 8(2):149–172, June 2000.

[158] Das I. and Dennis J. E. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. Structural and Multidisciplinary Optimization, 14:63–69, 1997.

[159] Scha_er David. Multiple objective optimization with vector evaluated genetic algorithms. In Proceedings of the 1st International Conference on Genetic Algorithms, pages 93–100, Hillsdale, NJ, USA, 1985. L. Erlbaum Associates Inc.

[160] Fonseca C.M. and Fleming P.J. Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. pages 416–423, 1993.

[161] Deb Kalyanmoy and Goldberg David. An investigation of niche and species formation in genetic function optimization. In Proceedings of the third international conference on Genetic algorithms, pages 42–50, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[162] Srinivas N. and Deb Kalyanmoy. Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation, 2:221–248, 1994.

[163] G¨unter Rudolph. Evolutionary search under partially ordered fitness sets. In Proceedings of the international Symposium on Information Science Innovations in Engineering of Natural and Artificial Intelligent Systems (ISI 2001, pages 818–822. ICSC Academic Press, 2001.

[164] Zitzler Eckart, Deb Kalyanmoy, and Thiele Lothar. Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation, 8:173–195, 2000.

[165] Sharma D., Kumar A., Deb K., and Sindhya K. Hybridization of sbx based nsga-ii and sequential quadratic programming for solving multi-objective optimization problems. pages 3003–3010, 2007.

[166] Zitzler E. and Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary Computation, 3:257 – 271, 1999.

[167] Coello Coello Carlos A., Lamont Gary B., and Veldhuizen David A. van. Evolutionary Algorithms for Solving Multi-Objective Problems. 2nd ed. Springer- Verlag, 2007.

[168] Wolpert D. and Macready W. No free lunch theorems for search. Technical Report SFI-TR-02- 010, Santa Fe Institute, 1995.

[169] Igel Christian and Toussaint Marc. A no-free-lunch theorem for non-uniform distributions of target functions. Journal of Mathematical Modelling and Algorithms, 2004.

[170] English T. M. Optimization is easy and learning is hard in the typical function. In Proceedings of the 2000 Congress on Evolutionary Computation: CEC00, volume 2, pages 924–931, 2000.

[171] Radcli_e Nicolas. The algebra of genetic algorithms. Annals of Mathematics and Artificial

Intelligence, 10:339–384, 1994.

[172] Watanabe Satosi. Knowing and Guessing: A Quantitative Study of Inference and Information. John Wiley and Sons Inc, 1969.

[173] Darrell Whitley. New insights about no free, 2004. http://www.cs.bham.ac.uk/~wbl/biblio/gecco2004/TUT025.pdf.

[174] English Thomas. No more lunch: Analysis of sequential search. In Congress on Evolutionary Computation, 2004. CEC2004, volume 1, pages 227 – 234. IEEE, Lubbock, TX, USA, 2004.

[175] Schumacher C., Vose M., and Whitley L. The no free lunch and problem description length. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001), pages 565–570, 2001.

[176] Christensen S. and Oppacher F. What can we learn from no free lunch? a first attempt to characterize the concept of a searchable function. In Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2001), 2001.

[177] Kennedy J. and Eberhart R. Swarm Intelligence. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2001.

[178] Poli R. An analysis of publications on particle swarm optimisation applications. Technical Report CSM-469, Department of Computer Science, University of Essex, UK, 2007.

[179] Poli R. Analysis of the publications on the applications of particle swarm optimisation. Journal of Artificial Evolution and Applications, pages 1–10, 2008.

[180] Shi Y. and Eberhart R. Parameter selection in particle swarm optimization. 7th International Conference on Evolutionary Programming., pages 591–600, 1998.

[181] Clerc M. and Kennedy J. The particle swarm Aˆ – explosion, stability and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation., 6(1):58–73, 2002.

[182] Van den Bergh F. An analysis of particle swarm optimizers. PhD thesis, Department Computer Science. University Pretoria. South Africa., 2002.

[183] Eberhart R. C., Simpson P., and Dobbins R. Computational Intelligence PC Tools. Academic Press Professional, 1996.

[184] Kennedy J. The particle swarm: Social adaption of knowledge. In International Conference on Evolutionary Computation, pages 303–308, Indianapolis, IN, USA, 1997.

[185] Clearwater S. H., Hogg T., and Huberman B. Cooperative problem solving. In Computation: The Micro and Macro View, pages 33–70, Singapore, 1992. World Scientific.

[186] Eshelman L. and Scha_er D. J. Real-coded genetic algorithms and interval schemata. In Foundations of Genetic Algorithms 2, pages 187–202, San Mateo, CA, 1992. Morgan Kaufmann.

[187] Goldberg D. E. Real-coded genetic algorithms, virtual alphabets, and blocking. Department of General Engineering, University of Illinois at Urbana-Champaign, 1990.

[188] Reynolds C.W. Flocks, herds and schools: A distributed behavioral model. In Computer Graphics, Proceedings of the ACM SIGGRAPH Conference, volume 21, pages 25–34, Anaheim, CA, USA, 1987.

[189] Shi Y. and Eberhart R. A modified particle swarm optimizer. In IEEE International Conference of Evolutionary Computation, Anchorage, Alaska, 1998.

[190] Shi Y. and Eberhart R. Empirical study of particle swarm optimization. In Proceedings of the Congress on Evolutionary Computation, pages 1945–1949, Washington D.C, USA, 1999.

[191] Alba E. and Dorronsoro B. The exploration/exploitation tradeo_ in dynamic cellular genetic algorithms. IEEE Transactions on Evolutionary Computation, 9(2):126–142, 2005.

[192] Clerc M. The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In Proceedings of the Congress on

Evolutionary Computation, pages 1951–1957, Washington DC, USA, 1999. IEEE Service Center.

[193] Clerc M. and Kennedy J. The particle swarm: Explosion, stability and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 6(1):58–73, 2002.

[194] pages 379–387, 1999.

[195] Eberhart R. and Shi Y. Comparing inertia weights and constriction factors in particle swarm optimization. In Proceedings of the Congress on Evolutionary Computing, pages 84–89, San Diego, USA, 2000. IEEE Service Center, Piscataway,NJ.

[196] Suganthan P. Particle swarm optimizer with neighbourhood operator. In Proceedings of the Congress on Evolutionary Computation, pages 1958–1961, Washington DC, USA, July 1999.

[197] Kennedy J. Small worlds and mega-minds: E_ects of neighbourhood topology on particle swarm performance. In Proceedings of the Congress on Evolutionary Computation, pages 1931–1938, Washington DC, USA, 1999. IEEE Service Center, Piscataway, NJ.

[198] Kennedy J. Stereotyping: Improving particle swarm performance with cluster analysis. In Proceedings of the Congress on Evolutionary Computing, pages 1507– 1512, San Diego, USA, 2000.

[199] Spears W. Simple subpopulation schemes. In Proceedings of the Evolutionary Programming Conference, pages 296–307, 1994.

[200] Lovbjerg M., Rasmussen T., and Krink T. Hybrid particle swarm optimiser with breeding and subpopulations. 2001.

[201] Kennedy J. and Spears W. Matching algorithms to problems: An experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. In Proceedings of the International Conference on Evolutionary Computation, pages 78–83, Anchorage, Alaska, 1998.

[202] Fukuyama Y. and Yoshida H. A particle swarm optimization for reactive power and voltage control in electric power systems. In Proceedings of the IEEE Congress on Evolutionary Computation, pages 87–93, Seoul, Korea, 2001.

[203] Lanzarini L., López J., Maulini J., and De Giusti A. A new binary pso with velocity control. In Advances in Swarm Intelligence - Second International Conference, ICSI 2011, pages 111–119, Chongqing, China, 2011. Springer-Verlag.

[204] Mojtaba Ahmadieh Khanesar. A novel binary particle swarm optimization. 18th Mediterranean Conference on Control and Automation, 2007.

[205] Sadri J. and Suen C. A genetic binary particle swarm optimization model. IEEE Congress on Evolutionary Computation, pages 656–663, 2006.

[206] Pampara G., Franken N., and Engelbrecht A. Combining particle swarm optimisation with angle modulation to solve binary problems. IEEE Congress on Evolutionary Computation, pages 89–96, 2005.

[207] Marandi A., Afshinmanesh F., Shahabadi M., and Bahrami F. Boolean particle swarm optimization and its application to the design of a dual-band dual-polarized planar antenna. IEEE Congress on Evolutionary Computation, pages 3212–3218, 2006.

[208] Huang V. L., Qin A. K., Deb K., Zitzler E., Suganthan P., Liang J. J., Preuss M., and Huband S. Problem definitions for performance assessment of multi-objective optimization algorithms. special session on constrained real-parameter optimization. Technical report, Nanyang Technological University, Singapore, 2007.

[209] Alba E. and Tomassini M. Parallelism and evolutionary algorithms. IEEE. Transactions on Evolutionary Computation, 6:443–462, 2002.

[210] Monson Christopher K. and Seppi Kevin D. The kalman swarm a new approach to particle motion in swarm optimization. In Genetic and Evolutionaru Computation - GECCO 2004, pages 140–150. Springer-Verlag, 2004.

[211] Van den Bergh F. An analysis of particle swarm optimizers. PhD thesis, Department Computer Science. University Pretoria. South Africa., 2002.

[212] Lanzarini L., Leza V., and De Giusti A. Particle swarm optimization with variable population size. Artificial Intelligence and Soft Computing – ICAISC 2008., 5097/2008:438–449, 2008.

[213] Kaisa Miettinen. Nonlinear multiobjective optimization, 1998.

[214] Rosenberg R. Simulation of genetic populations with biochemical properties, 1967.

[215] Scha_er David. Multiple objective optmization with vector evaluation genetic algorithms, 1984.

[216] Scha_er David. Multiple objective optmization with vector evaluation genetic algorithms, 1985.

[217] Scha_er David and Grefenstette John. Multi-objective learning via genetic algorithms, 1985.

[218] Knowles J. D. and Corne D. W. The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation. In In Congress on Evolutionary Computation (CEC99), volume 1, pages 98–105, Piscataway, NJ, 1999. IEEE Press.

[219] Hu Xiaohui, Shi Yuhui, and Eberhart Russ. Recent advances in particle swarm. In EDS Embeded System Group, Purdue University.

[220] Durillo, Garc´ıa Nieto, Nebro, Coello Coello, Luna, and Alba. Multi-objective particle swarm optimizers: An experimental comparison. In 5th International Conference on Evolutionary Multi-Criterion Optimization (EMO-2009). Springer, 2009.

[221] Abido. Two-level of nondominated solutions approach to multiobjective particle swarm optimization. In Genetic And Evolutionary Computation Conference Proceedings, pages 726 –733.

[222] Reyes Sierra and Coello Coello. Multi-objective particle swarm optimizers: A survey of the stateof-the-art. In International Journal of Computational Intelligence Research, pages 287–308, 2006.

[223] Mostaghim Sanaz and Teich Jurgen. Covering pareto optimal fronts by subswarms in multiobjective particle swarm optimization. In Congress on Evolutionary Computation, pages 1404–1404, 2004.

[224] Raquel Naval. An e_ective use of crowding distance in multiobjective particle swarm optimization. In Conference on Genetic and Evolutionary Computation, pages 257–264, New York, NY, 2005. GECCO.

[225] Reyes Sierra and Coello Coello. Improving pso-based multiobjective optimiza- tion using crowding, mutation and epsilon-dominance. In Evolutionary Multi- Criterion Optimization (EMO 2005), LNCS 3410, pages 505–519, 2005.

[226] Durillo J., Nebro A., Luna F., Dorronsoro B., and Alba E. jmetal: A java framework for developing multi-objective optimization metaheuristics. Technical report, Departamento de Lenguajes y Ciencias de la Computaci´on, University of M´alaga, E.T.S.I. Inform´atica, Campus de Teatinos, 2006.

[227] Deb K., Pratap A., Agarwal S., and Meyarivan T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Transactions on Evolutionary Computation, 6(2), 2002.

[228] Zitzler E., Deb K., and Thiele L. Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation, 8(2):173–195, 2000.

[229] Deb Kalyanmoy, Thiele Lothar, Laumanns Marco, and Zitzler Eckart. Scalable test problems for evolutionary multiobjective optimization. In Ajith Abraham, Lakhmi Jain, and Robert Goldberg, editors, Evolutionary Multiobjective Optimization, Advanced Information and Knowledge Processing, pages 105–145. Springer Berlin Heidelberg, 2005.

[230] Zitzler E. Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications. PhD thesis, Swiss Federal Institute of Technology (ETH) Zurich, 1999.

[231] Knowles J., Tiele L., and Zitzler E. A tutorial on the perfomance assessment of stochastic multiobjective optimizers.

[232] Zitzler E., Laumanns M., and Thiele L. Spea2: Improving the strength pareto evolutionary algorithm. In EUROGEN 2001, pages 95–100, Athens, Greece, 2002.

[233] Riget J. and Vesterstrom J. A diversity-guided particle swarm optimizer Aˆ – the arpso. EVALife Project Group Department of Computer Science, 2002.

[234] Akjiratikarl C., Yenradee P., and Drake P. Pso-based algorithm for home care worker scheduling in the uk. In Comput. Ind. Eng. 53, 4, 2007.

[235] Belmecheri F., Prins C., Yalaoui F., and Amodeo L. Particle swarm optimization to solve the vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows. In Parallel and Distributed Processing, Workshops and Phd Forum (IPDPSW). IEEE International Symposium, 2010.

[236] Brotcorne L., Laporte G., and Semet F. Ambulance location and relocation models. In European Journal of Operational Research 147, pages 451–463, 2003.

[237] Church R.L. and ReVelle C. The maximal covering location problem. In Papers of the Regional Science Association 32, pages 101–118, 1974.

[238] Dantzig G. and Ramser J. The truck dispatching problem. In Management Science, pages 80–91, 1959.

[239] Daskin M. A maximum expected covering location model: Formulation, properties and heuristic solution. In Transportation Science 17. 48-70, 1983.

[240] Fink E. Changes of problem representation: Theory and experiments. Springer-Verlag, Berlin, Germany, 2003.

[241] Garey M. R. and Johnson D. S. In Computers and Intractability: A Guide to the Theory of NPCompleteness. W.H. Freeman and Company, New York, 1979.

[242] Doerner K., Gutjahr W., Hartl R., Karall M., and Reimann M. Heuristic solution of an extended double-coverage ambulance location problem for austria. 2005.

[243] Gendreau M., Laporte G., and Potvin Y. Metaheuristics for the vrp. in the vehicle routing problem. In SIAM Monographs on Discrete Mathematics and Applications, pages 129–154. Philadelphia, 2002.

[244] Gendreau M., Laporte G., and Semet F. Solving an ambulance location model by tabu search. In Location Science 5, pages 75–85, 1997.

Esta edición de 150 ejemplares
se terminó de imprimir en Estudiocentro,
Bolívar, Buenos Aires, Argentina,
en el mes de abril de 2015.

Esta red se constituyó formalmente en noviembre de 1996 y actualmente 51 universidades argentinas son miembros activos. Sus objetivos son:
"Coordinar actividades académicas relacionadas con el perfeccionamiento docente, la actualización curricular y la utilización de recursos compartidos en el apoyo al desarrollo de las carreras de Ciencia de la Computación y/o Informática en Argentina".
"Establecer un marco de colaboración para el desarrollo de las actividades de posgrado en Ciencia de la Computación y/o Informática de modo de optimizar la asignación y el aprovechamiento de recursos".

Cuando hablamos de optimización en el ámbito de las ciencias de la computación hacemos referencia a la concreción de un objetivo utilizando la menor cantidad de recursos disponibles.
En el presente trabajo se estudia la utilización de metaheurísticas evolutivas para la optimización de problemas complejos, con uno y con más de un objetivo. En este trabajo se proponen cuatro variantes nuevas del algoritmo conocido como PSO (Particle Swarm Optimization). Tres de ellas aplicadas a problemas mono-objetivo, y una aplicada a problemas multiobjetivo.
Esta metaheuristica pertenece a la rama de Inteligencia de Enjambres (Swarm Intelligence). Estos métodos están inspirados en procesos biológicos y/o físicos, y tratan de simular el comportamiento propio de estos procesos. Estos tipos de algoritmos pertenecen a la categoria de Computación Evolutiva, cuyo representante mas difundido son los algoritmos genéticos.
Por último, se utilizan estas técnicas representativas del estado del arte en optimización multi-objetivo, aplicando las mismas a la problemática de la principal compañía de emergencias médicas y atención de consultas domiciliarias del país.