



UNIVERSIDAD DE DEUSTO

APLICACIÓN DE TÉCNICAS DE INTELIGENCIA ARTIFICIAL PARA LA PREDICCIÓN DE CONGESTIONES A CORTO PLAZO

Tesis doctoral presentada por Pedro López García
dentro del Programa de Doctorado en Ing. para la Sociedad de la Información y Desarrollo
Sostenible

Dirigida por Dr. Enrique Onieva Caracuel y
Dr. Asier Perallos Ruiz



UNIVERSIDAD DE DEUSTO

APLICACIÓN DE TÉCNICAS DE INTELIGENCIA ARTIFICIAL PARA LA PREDICCIÓN DE CONGESTIONES A CORTO PLAZO

Tesis doctoral presentada por Pedro López García
dentro del Programa de Doctorado en Ing. para la Sociedad de la Información y Desarrollo
Sostenible

Dirigida por Dr. Enrique Onieva Caracuel y
Dr. Asier Perallos Ruiz

El doctorando

El director

El director

Bilbao, Junio de 2016

Aplicación de técnicas de Inteligencia Artificial para la predicción de congestiones a corto plazo

Autor: Pedro López García

Directores: Enrique Onieva y Asier Perallos

Texto impreso en Bilbao

Primera edición, Junio de 2016

*Dedicado a todos aquellos que decidieron escalar una montaña sólo para saber
qué había al otro lado.*

Agradecimientos

Esta es, sin duda alguna, la sección más difícil de escribir de toda la tesis. Tengo que admitir que, cuando comencé este camino, no esperaba que trajera consigo tantos baches.

Por ello, en primer lugar me gustaría agradecer a *Enrique Onieva* su paciencia, su conocimiento y consejos, ya no sólo en lo que a trabajo se refiere, sino también en la vida. También a *Asier Perallos*, sin quien no habría empezado a escalar esta montaña. Sin él y su firme apuesta por mí, yo no habría empezado esta tesis.

A todo el grupo Mobility, quien me hizo sentirme como en casa desde el primer momento. Las ocho horas de todos los días serían mucho más duras sin las risas, los descansos, y todos esos cafés que nos salvan las tardes. Y aunque nunca escriba mi nombre en euskera, aquí no podría llamarme de otra manera. Los de Bilbao elegimos nacer donde nos da la gana. Especial mención al pasillo ciencia donde, aunque crean que sólo leemos y escribimos, también nos increpamos las revisiones de artículos, y lo que no son artículos. A *Maldini*, por cambiarme la tesis entera con sus revisiones y ser más que un co-autor en mis artículos y más que ser un compañero del día a día, ser un amigo. A *Luis*, por todas las preguntas que me ha hecho responder antes incluso de que me las hiciera el tribunal, y querer saber siempre cómo estoy, aunque yo nunca lo diga. A *Laura*, porque el inglés de este documento sería mucho peor de no ser por ella. A *Hugo*, por entender todos los movimientos de cabeza, y hacer que no añore mi acento del sur. A todos los que el espacio no me permite nombrar ampliamente: *Itziar, Browner, Ander, Pili, Idoia, Antonio, Bruno, Alfonso, Pablo, Unai, Leire, Nacho, Roberto, Gorka...* por ser familia en todos los momentos difíciles, e incluso en los buenos.

A mi *familia*, la de sangre y kilómetros. Nunca podré agradecer suficiente su apoyo incondicional, su espera infinita a que esté en casa, para disfrutarme poco tiempo y volver a dejarme marchar. Nunca han dicho "quédate", porque saben que no puedo. Se han alegrado de mis triunfos y me han aguantado durante las derrotas, que las ha habido. Han estado al otro lado del teléfono siempre. Se han despedido con una sonrisa incluso

más grande de la que tenían cuando volvía, para que no se me hiciera tan duro. A mi madre, por no perder nunca la paciencia y siempre tener una sonrisa en la voz. A mi padre, por enseñarme a no rendirme jamás. A mi **hermano**, por luchar incluso más que yo y recordarme que juntos somos más fuertes cuando más lo necesitaba. Espero que estéis tan orgullosos de mí como yo lo estoy de vosotros.

A todos los que me han apoyado, por estar. A todos los que se fueron, por enseñarme a quedarme. A todos los que pensaron que podría cuando yo pensaba que no. Esta tesis va por todos vosotros.

Pedro López

Junio de 2016

Abstract

Intelligent Transportation Systems (ITS) can be defined as the intersection between communication and electronic technologies with the aim of mitigating transports problems, whatever their type is. ITSs are applied to areas such as vehicular communications, traffic signal control systems, or road information systems.

ITSs are really important in road transport, whether for personal trips, or transport of goods or people. One of the main problems of road transport is the traffic congestion. Its early detection could help to take actions to decrease the noise, not only in urban streets but also in freeways; reduce the amount of polluting gases; to increase the performance and accuracy of road transport systems, and to save in public infrastructure for institutions. For all of this, the early detection of traffic congestion is a fundamental topic in ITS research field.

Along the last decade, and with the goal of giving a solution to this problem, autoregressive techniques, vehicular communication methods, and Soft Computing techniques have been applied. Focusing on the last ones, fuzzy logic and metaheuristics have emerged as good alternatives in the last years thanks to their adaptability and the easy way to represent the solutions they offer. In this context, a hybrid method that combines several metaheuristics has been developed in order to apply it to the optimization of fuzzy logic techniques for congestion forecasting. This algorithm counts with the ability of adapt itself to different problems, modifying its operators and internal configuration.

On the one hand, to validate the formulated hypothesis, traffic datasets have been created with real data. As a result, it is possible to evaluate the different configurations of the algorithm and its application to fuzzy systems. Also, the configurations are compared with themselves as well as with well-known techniques of the literature. With this comparative, the good performance of the proposed technique is proved. On the other hand, mathematical functions of different types and complexity have been used as benchmarks to prove the adaptability of the technique to other themes. Besides, an

in-deep study has been carried out in order to choose the parameters of the technique when it is applied to functions optimization.

Resumen

Los Sistemas Inteligentes de Transporte (ITS) se definen como la intersección de las tecnologías de la comunicación y la electrónica con el fin de mitigar los problemas del transporte, ya sea terrestre, marítimo o aéreo. Los ITS se aplican a áreas como las comunicaciones vehiculares, los sistemas de control de señales de tráfico, o los sistemas de información en carretera.

Los ITS tienen una gran importancia en el transporte por carretera, ya sean desplazamientos personales o transporte de mercancías o personas. Uno de los principales problemas del transporte terrestre es la congestión de tráfico. Su temprana detección puede ayudar a tomar medidas para disminuir el ruido, no sólo en entornos urbanos, sino también en autopistas; disminuir la cantidad de gases contaminantes emitidos; incrementar la eficacia y el rendimiento de los sistemas de transporte por carretera, y ahorrar en infraestructura pública para las instituciones. Por todo ello, la detección temprana de congestión es un tema fundamental en el campo de investigación de los ITS.

A lo largo de la última década, y con el objetivo de dar solución a este problema, se han utilizado técnicas autoregresivas, métodos basados en comunicaciones vehiculares, y técnicas de Soft Computing entre otras. Centrándonos en estas últimas, la lógica difusa y las metaheurísticas han surgido como fuertes alternativas en los últimos años gracias a su facilidad de adaptación y la representabilidad de las soluciones que proveen. En este contexto, se propone el desarrollo de un método híbrido combinando varias metaheurísticas de manera que, a través de la optimización de técnicas de lógica difusa, sea competitivo en ámbitos como la predicción de congestión de tráfico. Se busca, además, que dicho algoritmo se pueda adaptar a diferentes problemas, manteniendo de esta manera la adaptabilidad proporcionada por las metaheurísticas.

Con el objetivo de validar la hipótesis formulada, se han creado conjuntos de datos de tráfico reales donde poder evaluar las diferentes configuraciones del algoritmo y

su aplicación a sistemas difusos y poder compararlos, no sólo entre sí, sino con otras técnicas actuales de la literatura. De esta forma, se ha podido validar el buen rendimiento de la propuesta. Por otro lado, se han utilizado funciones matemáticas de optimización de diferentes tipos y complejidad para comprobar la adaptabilidad de la técnica y estudiar en profundidad los parámetros relacionados con la misma.

Índice general

Índice de figuras	xiii
Índice de tablas	xv
1 Introducción	1
1.1 Planteamiento del problema	2
1.2 Propósito e hipótesis	5
1.3 Motivación y objetivos	6
1.4 Intereses científicos y sociales	7
1.5 Metodología de investigación	9
1.6 Estructura del trabajo	11
2 Estado del Arte	13
2.1 Soft Computing	13
2.1.1 Metaheurísticas	14
2.1.2 Algoritmos Genéticos	16
2.1.3 Entropía Cruzada	19
2.1.4 Hibridación de técnicas	21
2.1.5 Lógica Difusa	24
2.1.5.1 Sistemas Basados en Reglas Difusas	26
2.2 Descripción de los Sistemas Inteligentes de Transporte	29
2.3 Optimización	32
2.3.1 Tipos de Optimización	32
2.3.2 Técnicas de Soft Computing aplicadas a Optimización	34
2.3.3 Funciones Benchmark	35

ÍNDICE GENERAL

3 Descripción del algoritmo	39
3.1 GACE: Descripción y Funcionamiento	39
3.2 Combinación de las técnicas: Clasificación y aportación	41
3.3 Sistemas Basados en Reglas Difusas: Aplicación	43
3.4 Optimización de funciones: Aplicación	45
3.5 Aspectos a tener en cuenta en la codificación del algoritmo	46
3.6 Implementación del algoritmo propuesto	46
4 Experimentación y resultados	51
4.1 Aplicación del algoritmo a problemas de predicción de tráfico	51
4.1.1 Datos reales de tráfico	52
4.1.1.1 Conjunto de datos	54
4.1.2 Codificación de las soluciones	57
4.1.3 Operadores	59
4.1.3.1 Operadores utilizados en el Algoritmo Genético	59
4.1.3.2 Operadores utilizados en la Entropía Cruzada	62
4.1.4 Configuración y resultados	63
4.2 Aplicación del algoritmo a funciones de optimización	78
4.2.1 Funciones utilizadas	79
4.2.2 Estudio de los parámetros del algoritmo	80
4.2.3 Comparativa de los resultados	84
5 Conclusiones y trabajo futuro	91
5.1 Difusión de los resultados	91
5.2 Conclusiones de la aplicación a la predicción de congestión	93
5.3 Conclusiones de la aplicación a funciones de optimización	94
5.4 Mejoras y líneas futuras de trabajo	95
A Conclusions and Future Work	99
A.1 Dissemination of the results	99
A.2 Conclusions about congestion forecasting	101
A.3 Conclusions about functions optimization	102
A.4 Improvements and future lines of work	103

ÍNDICE GENERAL

B Apéndice	107
B.1 Fórmulas de las funciones utilizadas	107
C Acrónimos	111
Bibliografía	113

Índice de figuras

1.1	Valores en porcentaje (eje X) de las respuestas a la pregunta <i>En un día normal, ¿qué método de transporte utiliza normalmente?</i> realizada en el Eurobarómetro de 2014 sobre Transporte.	2
1.2	Valores en porcentaje (eje X) de las respuestas a la pregunta <i>¿Cuál de los siguientes aspectos piensa que es uno de los problemas más serios que afectan a las carreteras?</i> realizada en el Eurobarómetro de 2014 sobre Transporte.	3
1.3	Metodología de investigación empleada	10
2.1	Clasificación de técnicas de Hard y Soft Computing	15
2.2	Ejemplo de funciones de pertenencia triangulares y trapezoidales	25
2.3	Ejemplo de un sistema basado en reglas difusas	26
2.4	Ejemplo de un sistema de tipo TSK	27
2.5	Tipos de jerarquías de FRBS: HFRBS en serie.	28
2.6	Tipos de jerarquías de FRBS: HFRBS en paralelo.	28
2.7	Tipos de jerarquías de FRBS: HFRBS híbrido.	28
2.8	Diagrama resumen de los Sistemas Inteligentes de Transporte, sus áreas y sus aplicaciones	32
2.9	Artículos publicados durante el año 2014 en cada tipo de optimización (fuente: Scopus)	34
3.1	Pasos que sigue el algoritmo GACE	42
4.1	Página de Caltrans Performance Measurement System (PeMS)	52
4.2	Segmento de la carretera I5. Los sensores se denotan por S , las rampas de salida por RS y las de entrada por RE	53

ÍNDICE DE FIGURAS

4.3	Dataset Punto Simplificado. Sólo tiene en cuenta los sensores $S1$, $S7$ y $S13$, y una combinación de las rampas de entrada y salida antes y después del punto de interés $S7$.	54
4.4	Ejemplo de codificación 'lateral tuning' para cuatro MFs. Cada X_i puede tomar un valor en el intervalo $[-1, 1]$.	58
4.5	Ejemplo de codificación de un PHFRBS con seis variables de entrada.	60
4.6	Variante del cruce de orden usada en el trabajo. Primero, se elige el punto de corte. Posteriormente, se selecciona el orden y finalmente se crean los nuevos individuos.	61
4.7	Fases de creación de la parte $C_{jerarquia}$ en un nuevo individuo.	63
4.8	Transformación de vector de jerarquía a vector de orden y viceversa	64
4.9	Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DP_5 (arriba), DP_{15} (centro), y DP_{30} (abajo).	69
4.10	Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DPS_5 (arriba), DPS_{15} (centro), y DPS_{30} (abajo).	70
4.11	Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DS_5 (arriba), DS_{15} (centro), y DS_{30} (abajo).	71
4.12	Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DSS_5 (arriba), DSS_{15} (centro), y DSS_{30} (abajo).	72
4.13	Resultados del Test de Student para los datasets DP .	74
4.14	Resultados del Test de Student para los datasets DPS .	74
4.15	Resultados del Test de Student para los datasets DS .	75
4.16	Resultados del Test de Student para los datasets DSS .	75
4.17	Ranking promedio de cada dupla de valores utilizada en los posibles valores de dimensión dim . El eje x contiene los porcentajes posibles de p_{ga} mientras que en el eje y se exponen los valores del porcentaje p_{up} . En la parte superior se muestran los gráficos de calor de $dim = 5$ (izquierda) y $dim = 10$ (derecha). En la parte inferior, los correspondientes a $dim = 20$ (izquierda) y $dim = 40$ (derecha).	83

Índice de tablas

1.1	Actividades específicas que quedan englobadas en los objetivos impuestos para el cumplimiento de la hipótesis	8
2.1	Tabla de artículos en la literatura de Sistemas Jerarquicos Basados en Reglas. . . .	29
2.2	Tipos de benchmark encontrados en la literatura y sus características	37
4.1	Valores de congestión y su cálculo.	53
4.2	Clasificación de los datasets según el número de sensores utilizados y el cálculo de la congestión	55
4.3	Datasets Punto y Punto Simplificado. Los valores antes de la fecha indican su valor original, mientras que los que aparecen después se obtienen tras la reducción. Cada columna indica el tipo de congestión.	56
4.4	Datasets Sector y Sector Simplificado. Los valores antes de la fecha indican su valor original, mientras que los que aparecen después se obtienen tras la reducción. Cada columna indica el tipo de congestión.	56
4.5	Valores iniciales por cada una de las partes de \bar{x} y σ	62
4.6	Media de MAE y desviación en los conjuntos de prueba de cada una de las técnicas para datasets Punto y Punto Simplificado	65
4.7	Media de MAE y desviación en los conjuntos de prueba de cada una de las técnicas para datasets Sector y Sector Simplificado	66
4.8	Media del error sMAPE en los datasets DP y DPS por cada una de las técnicas. .	66
4.9	Media del error sMAPE en los datasets DS y DSS por cada una de las técnicas. .	67
4.10	Comparativa de las configuraciones de GACE con los algoritmos del estado del arte	68
4.11	Porcentaje de veces en los que los resultados de un algoritmo son mejores que los obtenidos por otro (+)	73

ÍNDICE DE TABLAS

4.12	Porcentaje de veces en los que los resultados de un algoritmo son significativamente mejores que los obtenidos por otro (++)	74
4.13	Media de posición para cada técnica cuando predice diferentes niveles de congestión. Los dos mejores valores están resaltados para cada caso.	76
4.14	Mejores configuraciones de GACE que predicen congestión para cada conjunto de datos.	76
4.15	Valores de los diferentes parámetros utilizados en la experimentación	80
4.16	Ranking promedio de los diferentes algoritmos utilizados por dimensión y valor de p_{ga}	81
4.17	Valores de los diferentes parámetros utilizados en la segunda parte de la experimentación	82
4.18	Parámetros utilizados para la comparativa entre el algoritmo y los métodos del estado del arte tras los estudios realizados	84
4.19	Media del error de las técnicas para $dim = 5$	86
4.20	Error medio de las técnicas para $dim = 10$	87
4.21	Error medio de las técnicas para $dim = 20$	88
4.22	Error medio de las técnicas para $dim = 40$	89
4.23	Resultados del test de Friedman para cada dimensión	90
4.24	Estadísticas de los test de Holm y Finner para cada dimensión y técnica	90
B.1	Funciones Separables: Nombre y fórmulas.	107
B.2	Funciones con condicionamiento bajo o moderado: Nombres y funciones.	108
B.3	Funciones con condicionamiento elevado y unimodales: Nombres y fórmulas.	108
B.4	Funciones multimodales con estructura global adecuada: Nombres y fórmulas.	108
B.5	Funciones multimodales con estructura global débil: Nombres y fórmulas.	109

La gente prefiere olvidar lo imposible; les hace la vida más fácil.

Neil Gaiman

1

Introducción

El transporte y la movilidad juegan un papel fundamental en nuestra vida diaria. Acciones como viajar, el transporte de mercancías o pasajeros, o incluso ir al trabajo forman parte de nuestro día a día. La mayoría de los desplazamientos que realizamos tienen como medio la carretera, como se puede apreciar en los datos aportados por el último eurobarómetro sobre la calidad del transporte emitido por la Comisión Europea [Commission 14]. De este estudio, podemos extraer como primera conclusión que el método más utilizado para la mayoría de nuestros desplazamientos diarios es, sin duda alguna, el coche particular. La Figura 1.1 muestra la gran diferencia existente entre el uso de este medio de transporte en comparación con el resto en lo que al día a día se refiere, realizando más de la mitad de los desplazamientos en este tipo de transporte. A estos datos hay que añadir que el coche es, por encima de medios de transporte como el avión o el tren, el más utilizado junto a la autocaravana en los desplazamientos de larga distancia (al menos, 300 km). Centrándonos en el transporte por carretera por su amplio uso, en términos de calidad, un 38 % de los encuestados opina que ha mejorado en los últimos 5 años en su país. A la par, un 25 % cree que se ha deteriorado, mientras que el 11 % opina que se ha deteriorado mucho.

Siguiendo con el hilo sobre el transporte por carretera y sus posibles inconvenientes, a la pregunta de '¿cuál de los siguientes eventos supone un mayor problema a la hora de la circulación por carretera?', y siendo posible un máximo de tres respuestas, un 60 % de los encuestados eligió la congestión de tráfico como una de sus opciones, seguido por el mantenimiento de la carretera con un 59 %. El resto de respuestas así como sus porcentajes se muestran en la Figura 1.2.

1. Introducción

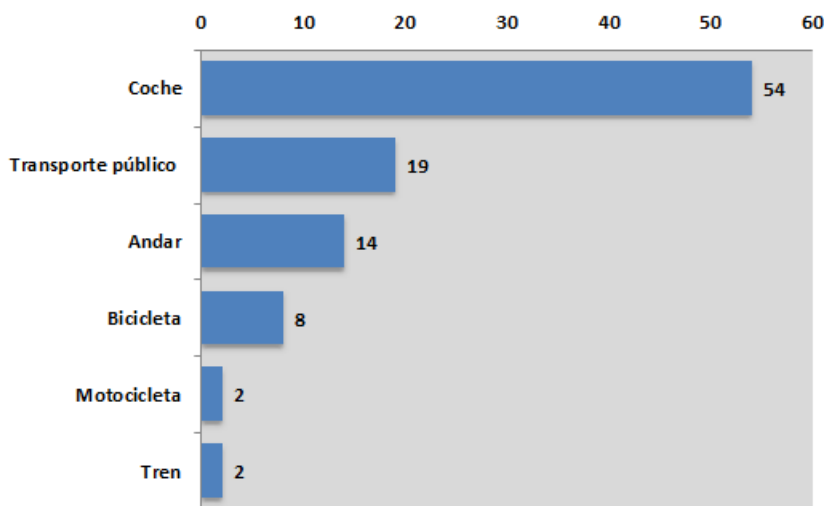


Figura 1.1: Valores en porcentaje (eje X) de las respuestas a la pregunta *En un día normal, ¿qué método de transporte utiliza normalmente?* realizada en el Eurobarómetro de 2014 sobre Transporte.

Los Sistemas Inteligentes de Transporte, del inglés *Intelligent Transportation Systems* (ITS), se definen como la intersección de las tecnologías de la comunicación y la electrónica con el fin de mitigar los problemas del transporte, ya sea terrestre, marítimo o aéreo. Los ITS se aplican en áreas como las comunicaciones vehiculares o los sistemas de control de señales de tráfico. Otros ejemplos de aplicación son los sistemas de información en carretera o las cámaras que detectan la velocidad o la matrícula de un vehículo automáticamente.

Por otro lado, la congestión del tráfico y su temprana detección es un tema fundamental en el campo de investigación de los ITS. Uno de los objetivos en este ámbito es realizar una predicción satisfactoria en entornos urbanos y carreteras. Llevar a cabo con éxito dicha tarea puede ayudar a tomar medidas para, por ejemplo, disminuir el ruido, no sólo en entornos urbanos, sino también en autopistas; ahorrar energía, lo que se complementa con un descenso de la polución; incrementar la eficacia y el rendimiento de los sistemas de transporte, y ahorrar en infraestructura pública para las instituciones.

1.1 Planteamiento del problema

A lo largo de la última década, se han propuesto y desarrollado diferentes métodos de cara a realizar predicciones del estado futuro del tráfico de manera satisfactoria. Dos de los métodos más usados en este periodo de tiempo han sido el Filtro de Kalman, del inglés *Kalman Filter* (KF) [Zhang 12b]

1.1 Planteamiento del problema



Figura 1.2: Valores en porcentaje (eje X) de las respuestas a la pregunta *¿Cuál de los siguientes aspectos piensa que es uno de los problemas más serios que afectan a las carreteras?* realizada en el Eurobarómetro de 2014 sobre Transporte.

y *Autoregressive Integrated Moving Average* (ARIMA) [Tan 07]. El KF utiliza un conjunto de ecuaciones que proporcionan un medio de cálculo eficiente para estimar el estado del proceso de forma que minimice el error medio [Welch 95]. Este filtro tiene sus ventajas en varios aspectos: mantiene estimaciones pasadas, presentes e incluso de futuros estados. Además de todo esto, es capaz de hacerlo cuando el modelo del sistema es desconocido. Por otro lado, ARIMA es una generalización del modelo ARMA (*AutoRegressive Moving Average*) [Box 13], adaptado para series temporales de manera que pueda predecir puntos futuros en dichas series. Sin embargo, estos procesos tienen varios problemas. Para los KFs tradicionales, la precisión de la predicción puede ser inestable [Wang 05]. Por otro lado, en el caso de ARIMA, el método puede ser incapaz de predecir correctamente series temporales de flujo de tráfico dado que están basados en teoría de procesos estocásticos estacionarios en los que normalmente se realizan supuestos [Tan 07].

No sólo se han desarrollado los modelos anteriores si no que, a lo largo de los últimos años, aparecen otras alternativas como pueden ser las comunicaciones Vehículo a Vehículo (V2V) en autopistas de largo recorrido [Bauza 13] o el esquema llamado Time-of-Day (TOD) [Jia 06], donde se divide un día en varios intervalos y se intenta encontrar los controladores óptimos para cada uno de ellos.

1. Introducción

Por otro lado, otro enfoque para la predicción del estado del tráfico es el que aportan las técnicas de Soft Computing como Máquinas Vector Soporte, del inglés *Support Vector Machines* (SVM), Redes Neuronales, del inglés *Neural Networks* (NN), Algoritmos Genéticos, del inglés *Genetic Algorithms* (GA) o Sistemas basados en Reglas Difusas, del inglés *Fuzzy Rule-Based Systems* (FRBS), ya sea para predecir tanto la fluidez del tráfico, como el tiempo de viaje, entre otras variantes. Estas técnicas se han usado por separado, o combinadas entre sí de cara a paliar sus posibles debilidades o cimentar los puntos fuertes de las mismas.

Por otra parte, la aplicación de métodos probabilísticos como el método de la Entropía Cruzada, del inglés *Cross Entropy* (CE) [Rubinstein 99] o el método de Adaptación de la Matriz de Covarianza, del inglés *Covariance Matrix Adaptation* (CMA) [Hansen 01] en este tipo de problemas se contempla en artículos como [Lebacque 09] o [Stellet 15].

Prestando atención en primer lugar a las técnicas de Soft Computing, las SVMs se encuentran con dificultades para obtener valores precisos cuando tratan con una gran cantidad de variables [Wang 13]. En el caso de las NN, éstas obtienen buenos resultados y han atraído más atención a lo largo de los años. Sin embargo, debido al problema de óptimo local, donde la solución queda estancada en un valor que, a pesar de ser óptimo, no es el mejor resultado posible, y su poca habilidad de generalización, su eficacia es, en algunos casos, limitada [Posawang 10]. Además, el resultado de una NN depende no sólo del entrenamiento de la red sino de la cantidad de datos de calidad disponibles y de los parámetros definidos. Por su parte, los métodos basados en poblaciones, como son los GA tienen problemas a la hora de explotar el espacio de búsqueda, siendo su fuerte la exploración del mismo [Talbi 02]. En el caso de los métodos probabilísticos, es importante la manera en la que se crean sus individuos, a la vez que su falta de exploración del espacio de búsqueda en el que se encuentran, siendo su principal baza la explotación de éste.

Con la idea ya no sólo de paliar las debilidades de cada técnica sino de combinar las partes en las que estas destacan, surgen los algoritmos híbridos. La hibridación se puede definir, de manera breve, como la combinación de técnicas de manera que se cree una sinergia entre ellas. De las técnicas anteriormente mencionadas, existen trabajos en el ámbito de los ITS que combinan, por ejemplo, GAs con FRBS [Zhang 14], NN con lógica difusa [Zargari 12], e, incluso, ARIMA con SVM [Zhang 12b].

La aplicación de todas estas técnicas al ámbito de los ITS en general y a la predicción del estado del tráfico en particular, así como la importancia de realizar esta última lo más satisfactoriamente posible convierten a este tema, ya no sólo en un tema de actualidad, sino en uno de vital importancia de cara al futuro de las comunicaciones por carretera.

A su vez, todas las técnicas de las que se ha hablado durante esta introducción son utilizadas en muchos y variados ámbitos. Uno de ellos, la optimización, ya sea de problemas del mundo real [Lu 14, DellÁmico 15] como creados artificialmente [Syberfeldt 09, Wang 09], es un tema que siempre ha sido bien recibido y genera una creciente atención en la comunidad científica, como se demuestra en futuras secciones. El reto que supone resolver los problemas antes especificados es una motivación para su estudio, sobre todo en el área de la inteligencia artificial. Hay muchos tipos de optimización, entre los que destacan la optimización numérica [Schwefel 81], lineal [Bertsimas 97], continua [Eiselt 87] o combinatoria [Papadimitriou 98]. La creación de técnicas de manera que sean capaces de adaptarse a ámbitos muy diferentes con el mínimo de cambios posibles en su estructura es un aliciente para la creación de los diferentes tipos de métodos, ya sean aplicados en solitario o realizando cualquier tipo de hibridación.

1.2 Propósito e hipótesis

Como se ha expuesto en el apartado anterior, poder prever la formación de congestiones en autopistas y autovías a corto plazo es un reto para los ITS. Realizar dicha predicción correctamente conllevaría, entre otros aspectos, la disminución de la polución, el ruido y el tiempo de viaje así como un aumento de la productividad en el transporte de mercancías y personas.

Uno de los propósitos de la tesis presentada en esta memoria está motivado por la intención de predecir las posibles retenciones a corto plazo que puedan darse en todo tipo de carreteras. Una de las ventajas de realizar dicha predicción a corto plazo es la adaptación de la toma de decisiones al momento exacto en el que ocurren los sucesos. Por ejemplo, si se produce un accidente en la carretera por la que circulamos, es posible que se produzca una retención al poco tiempo. En el caso de realizar esta predicción a largo plazo, no se podrían prever las posibles situaciones inesperadas en las carreteras. Otra de las ventajas es la capacidad de organizar un desplazamiento antes de comenzar. Si el usuario conoce la probabilidad con la que existirían retenciones en la vía que forma parte de su ruta, podrá organizar, incluso durante el desplazamiento, la ruta que le lleve a utilizar el menor tiempo posible para llegar a su destino. Esto, aunque también se puede llevar a cabo con una predicción a largo plazo, lleva a que el usuario no se tenga que preocupar por su ruta hasta el momento anterior de comenzarla. Para llevar a cabo dicho propósito, no sólo se ha trabajado en la predicción de la congestión en un único punto sino a lo largo de toda la sección de una autopista, dando un nuevo enfoque a este tipo de problema.

A su vez, se busca que lo desarrollado durante esta tesis no sea sólo aplicable a un campo concreto, sino que sea capaz de adaptarse a la optimización en otros campos diferentes de la zona de

1. Introducción

confort en la que nos movemos, siendo esta los ITS y su aplicación a la predicción congestiones de tráfico.

De manera más concreta, se propone el desarrollo de un Algoritmo Híbrido, combinando Métodos Probabilísticos con Algoritmos Evolutivos, para la optimización de un tipo de Sistemas Basados en Reglas Difusas llamado Sistemas Basados en Reglas Difusas Jerárquicos, del inglés *Hierarchical Fuzzy Rule-Based Systems* (HFRBS), que será aplicado al caso de la congestión de tráfico. Dicho algoritmo será, a su vez, capaz de optimizar funciones matemáticas continuas de complejidad creciente con un rendimiento óptimo.

Teniendo en cuenta lo anterior, y dados los propósitos expuestos, la hipótesis que se quiere demostrar con la realización de este trabajo es la siguiente:

La hibridación de Algoritmos Evolutivos y Métodos Probabilísticos permitirá desarrollar métodos más competitivos en ámbitos como la predicción de congestión de tráfico, a través de la optimización de jerarquías de Sistemas Basados en Reglas Difusas, y la optimización de funciones continuas.

1.3 Motivación y objetivos

La motivación sobre la que se ha edificado esta tesis es la de crear un algoritmo híbrido que sea capaz, no sólo de ayudar a los sistemas basados en reglas difusas a predecir de manera lo más fiable posible las congestiones de tráfico que se puedan producir en entornos urbanos o carreteras, sino que se pueda aplicar a otro tipo de problemas diferentes, como pueden ser problemas de optimización, previo paso del estudio del problema en cuestión y adaptación de sus características a dicho problema.

De una manera más formal, y siendo más concretos, el trabajo que nos ocupa tiene como principal objetivo la utilización de técnicas híbridas combinando Algoritmos Evolutivos como los GAs con Métodos Probabilísticos como la CE para la optimización de HFRBS y los sistemas que lo forman, de cara a predecir con el menor error posible los futuros estados del tráfico en una carretera. Además, como objetivo complementario, el algoritmo desarrollado debe adaptarse de manera satisfactoria a otro tipo de problemas de optimización, como es el caso de los problemas de optimización de funciones matemáticas.

Ampliando el nivel de detalle de estos objetivos, podemos encontrarnos con los siguientes objetivos específicos:

- ◊ O1: Diseño de la técnica para la combinación de los algoritmos de manera óptima. Dicho objetivo quedará cumplido tras el estudio en profundidad de las técnicas que forman la

hibridación y la creación de sinergia entre ellas, aumentando sus virtudes y paliando sus posibles debilidades.

- ◇ O2: Configuración de la hibridación de cara a su utilización para la optimización de los FRBSs. Este punto resulta crucial, dado que la elección de los operadores de las diferentes partes que forman el algoritmo desarrollado determina, en parte, el rendimiento de cara a la solución de los problemas presentados.
- ◇ O3: Creación del entorno de pruebas. El estudio del estado del arte relacionado con la temática de predicción es indispensable para realizar una experimentación amplia y de calidad. En el caso de que en dicho estado del arte no se encuentren conjuntos de datos ya creados para el estudio de la predicción de tráfico, estos deberán crearse, a poder ser, a partir de datos reales.
- ◇ O4: Estudio y evaluación de los resultados. Relacionado con el objetivo O3, se aplicará a los conjuntos de datos la jerarquía de sistemas y, a su vez, el algoritmo a estos, de cara a su optimización. Durante dicha aplicación, es necesario un estudio de los parámetros de la propuesta, de cara a encontrar aquellos con los que se obtenga un mejor rendimiento. Además, los resultados obtenidos deberán cotejarse con algoritmos del estado del arte de la misma índole, de cara a obtener una comparativa justa y objetiva.
- ◇ O5: Adaptación de la técnica a problemas de optimización de funciones matemáticas. Una vez realizado el estudio anterior, y siguiendo el esquema obtenido con el cumplimiento del objetivo O1, la técnica propuesta deberá adaptarse a la optimización de funciones. Esto conllevará el estudio de sus parámetros, a la vez que la utilización de las configuraciones obtenidas en el objetivo O4. En este apartado también se incluye la creación u obtención del entorno de pruebas correspondiente.

El llevar a cabo estos objetivos terminará en el cumplimiento del objetivo tanto primario como secundario expuesto en este apartado. De cara al desarrollo de esta tesis, la Tabla 1.1 expone actividades más específicas que quedarían englobadas dentro de estos objetivos.

1.4 Intereses científicos y sociales

Una vez introducido el ámbito de los problemas a tratar, y tras describir la motivación, propósitos y la hipótesis a cumplir, así como los objetivos que nos llevarán a la validación de la misma, este apartado pretende destacar los intereses tanto científicos como sociales que lleva la realización de este trabajo.

1. Introducción

Actividad	Objetivos
Análisis de las técnicas utilizadas en la hibridación, así como sus operadores y parámetros	O1 y O2
Análisis del estado del arte en metaheurísticas e hibridación	O1
Análisis de los diferentes tipos de FRBS así como su aplicación al problema descrito	O2
Obtención de los datos a utilizar, así como su tratamiento en caso de ser necesario	O3 y O4
Aplicación de la solución propuesta y evaluación comparativa con diferentes métodos de la literatura	O4
Estudio de la optimización de funciones así como obtención de las más utilizadas en la literatura	O5
Estudio de los parámetros y operadores para la adaptación del algoritmo a problemas de optimización de funciones	O5
Aplicación del algoritmo y estudio de su configuración para optimización de funciones	O5

Tabla 1.1: Actividades específicas que quedan englobadas en los objetivos impuestos para el cumplimiento de la hipótesis

Se busca, por tanto, destacar por qué es necesario realizar esta tesis y todo lo que conlleva que se cumplan los objetivos expuestos en el apartado anterior.

Teniendo en cuenta esto, el interés social del tema que se aborda quedaría destacado en los siguientes puntos:

- ◇ Predecir con éxito el estado del tráfico se torna básico para el futuro de las comunicaciones por carretera, ya no sólo para los conductores que día a día utilizan sus vehículos propios para los desplazamientos por carretera, sino también para las empresas de transporte, cuyos beneficios y satisfacción de sus clientes dependen en gran medida del tiempo que tardan en realizar sus servicios, ya sea transporte de personas o paquetería, entre otras. Las técnicas desarrolladas en esta tesis buscan ser capaces de informar con el menor error posible sobre el estado de la congestión en una vía, ya sea en un punto concreto de la misma o en el sector de carretera que nos sea de interés. De esta manera, el conductor podrá organizar su desplazamiento antes de comenzar, o incluso durante el mismo, de cara a evitar las posibles congestiones que se puedan dar.
- ◇ Ligado con lo anterior, ser capaces de elegir la ruta dependiendo del factor de congestión de la vía hará que el transporte, ya sea personal, de pasajeros o de paquetería, sea más eficiente. Esto conlleva, a su vez, que la congestión que se predice disminuya, o desaparezca más deprisa. Dicha disminución tiene, de la misma manera, una serie de ventajas, de las cuáles destacan las siguientes:
 - Disminución del tiempo requerido para el desplazamiento, lo que minimiza el riesgo de accidentes.

- Disminución del consumo tanto energético como de carburante. Esta acción conlleva a su vez el ahorro para el conductor o la empresa por tener que repostar con menos frecuencia, a la vez que reducir el coste del hidrocarburo.
 - Reducción de emisiones de CO₂. Dada la importancia y compromiso por parte de la sociedad y los gobiernos para la reducción de gases contaminantes a la atmósfera, esta ventaja se convierte en vital. La mencionada disminución de combustible, y la reducción del tiempo que se está en carretera ayuda a su vez a reducir la emisión de CO₂.
 - Decremento de la contaminación acústica tanto en entornos urbanos, aumentando de esta manera la calidad de vida, como en autopistas.
- ◊ Por otro lado, el querer realizar dicha predicción a corto plazo conlleva que el individuo pueda adaptarse a los problemas en el momento exacto en el que ocurren los sucesos que los crean. Por ejemplo, si ocurre un accidente y comienza a crearse una retención, los conductores, avisados de ello, pueden cambiar su ruta en el instante, lo que ayudaría además a que los servicios sanitarios o de seguridad lleguen antes al lugar del suceso al no encontrarse, prácticamente, con tráfico en su camino.

Se ha mostrado a grandes rasgos durante este capítulo la importancia que tienen los ITS en la última década, aportando técnicas de diferente índole de cara a encontrar posibles soluciones a los problemas planteados en esta temática. A su vez, la comunidad científica, en forma de contribuciones a revistas y congresos, aportan sus puntos de vista con respecto a este tema. En capítulos posteriores de esta tesis, se mostrará con mayor detalle el estado del arte relacionado ya no sólo con los ITS sino con la predicción de congestión.

Para finalizar esta sección, y no por ello siendo menos importante, la optimización de funciones matemáticas conlleva un interés científico que se remonta al comienzo de la computación en sí misma. Funciones complejas se pueden resolver o encontrar su óptimo en un tiempo razonable gracias al uso de diferentes algoritmos, ya sean probabilísticos o evolutivos. De esta manera, desarrollar un algoritmo capaz de adaptarse a este tipo de problemas, no siendo esa su función principal, tiene un gran interés académico, sobre todo si los resultados obtenidos alcanzan, o al menos se acercan en gran medida, al óptimo de la función.

1.5 Metodología de investigación

El campo de los ITS en general y la predicción del estado del tráfico en particular es un campo que, en la última década, ha generado cada vez un mayor número de producción científica. Por tanto,

1. Introducción

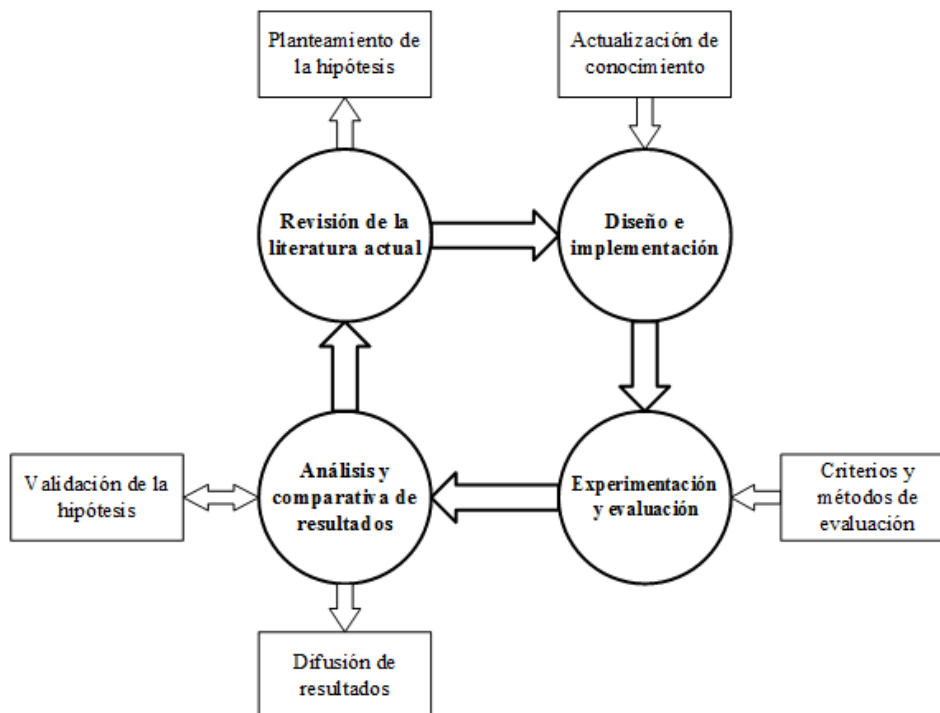


Figura 1.3: Metodología de investigación empleada

se hace necesario acogerse a una metodología de investigación de ciclo continuo, donde actualizar el conocimiento adquirido y mejorar en la mayor medida de lo posible la solución desarrollada se erigen como pilares básicos.

De esta manera, se ha planteado un proceso iterativo en el que la terminación de cada ciclo conlleva el refinamiento de la solución planteada para cumplir la hipótesis. La idea básica tras este procedimiento es que los conocimientos adquiridos en su fase inicial ayuden a desarrollar una técnica cada vez más prometedora, siendo capaz de mejorar tanto los resultados obtenidos por los métodos más actuales como el concepto de los mismos, aportando originalidad y adaptabilidad entre otras características.

La Figura 1.3 muestra la metodología de investigación empleada, cuyas fases pueden resumirse de la siguiente manera:

- ◇ *Revisión de la literatura actual.* En esta fase, se investiga el estado del arte relacionado con los campos en los que se está desarrollando el trabajo, haciendo hincapié en la predicción de congestión y algoritmos híbridos. Para lograr esto, se hará uso de la bibliografía relacionada, centrándose en publicaciones en revistas científicas de nivel, al igual que en congresos, tanto

nacionales como internacionales. El conocimiento adquirido en esta fase desemboca en el planteamiento de la hipótesis, tras localizar y analizar las posibles mejoras a realizar.

- ◇ *Diseño e implementación.* Tras la adquisición o actualización del conocimiento en la fase anterior, y siempre sobre la hipótesis planteada, en esta fase se diseña (o modifica si el ciclo no es el primero) e implementa la solución que se propone para intentar probar la hipótesis.
- ◇ *Experimentación y evaluación.* El objetivo de esta fase es el de probar la solución desarrollada en el paso anterior siguiendo un proceso de experimentación y evaluación. En dicho proceso, resulta básico aportar unos criterios y métodos de evaluación, en los que se incluyen los conjuntos de datos empleados (o las fórmulas en el caso de la optimización de funciones), la parametrización utilizada o las técnicas del estado del arte con las que se compararán los resultados en una fase posterior.
- ◇ *Análisis y comparativa de resultados.* Tras el paso por la fase anterior, los resultados obtenidos tienen que ser analizados con las métricas necesarias y comparados con los obtenidos por otras técnicas actuales de la literatura. Una vez realizado esto, se debe comprobar si la hipótesis planteada ha quedado contrastada. En tal caso, se podrá dar por finalizado el desarrollo de la tesis, teniendo siempre en cuenta las posibles actualizaciones del estado del arte. Esta etapa debe dar también como resultado la difusión de los resultados, ya sea tanto en congresos como en artículos de revista científicas.

1.6 Estructura del trabajo

La presente memoria se estructura en los siguientes capítulos:

- ◇ El Capítulo 2 presenta un estudio del estado del arte en las diferentes materias de las que se nutre este trabajo, en especial en la Soft Computing y sus componentes, los Sistemas Inteligentes de Transporte, y los tipos de Optimización, así como sus técnicas y funciones.
- ◇ La descripción del algoritmo propuesto en esta tesis, así como su funcionamiento, aportación de la hibridación de las técnicas y aplicación a los FRBS y el resto de problemas se encuentra en el Capítulo 3.
- ◇ En el Capítulo 4 se detallan las diferentes experimentaciones y los resultados obtenidos en cada una de ellas, así como los operadores y codificaciones utilizados por el algoritmo para cada tipo de problema.

1. Introducción

- ◇ Por último, las conclusiones extraídas del trabajo realizado y los trabajos futuros que se realizarán tras la terminación de esta tesis se recogen en el Capítulo 5.

*Dicen que la curiosidad mató al gato, pero no cuentan
si lo que descubrió merecía la pena.*

José Saramago

2

Estado del Arte

En este capítulo se recoge el estado del arte en los diferentes ámbitos en los que se desarrolla este trabajo. En la Sección 2.1 se recoge diferente información sobre la Soft Computing, con especial hincapié en metaheurísticas y su hibridación, y Lógica Difusa, ampliando información sobre los FRBS. Por otro lado, en la Sección 2.2 se encuentran los Sistemas Inteligentes de Transporte, con diferentes definiciones dadas por instituciones gubernamentales y tipos en los que se clasifican. Por último, en la Sección 2.3 se definen la optimización de funciones, explicando los tipos que nos podemos encontrar, técnicas que se han aplicado a este tema, y repositorios de funciones que se encuentran en la literatura actual de cara a evaluar los algoritmos de manera fiable y justa.

2.1 Soft Computing

Zadeh definió en [Zadeh 94] la Soft Computing como un conjunto de metodologías cuyo objetivo es explotar la tolerancia al error y la incertidumbre para conseguir manejabilidad, robustez, soluciones de bajo coste y mejores representaciones de la realidad.

Podemos encontrar una definición más reciente en [Verdegay 08], donde se refiere a la Soft Computing como un conjunto de técnicas y métodos que permiten tratar las situaciones prácticas reales de la misma forma que suelen hacerlo los seres humanos, es decir, en base a inteligencia, sentido común, consideración de analogías, aproximaciones, etc. Por tanto, se puede considerar a la *Soft Computing* lo contrario a lo que se denomina *Hard Computing*.

2. Estado del Arte

La Figura 2.1 muestra una diferenciación basándose en los métodos que utilizan tanto la Hard como la Soft Computing. Siguiendo dicha figura, la Hard Computing engloba los modelos precisos, como son el razonamiento inductivo y la optimización numérica tradicional. Estos métodos requieren un modelo analítico y a menudo una gran cantidad de tiempo de cómputo. A su vez, son métodos determinísticos en los que los datos proporcionados deben estar completos para obtener una solución viable. Por su lado, la Soft Computing trata modelos aproximados, jugando a la par con la imprecisión, la incertidumbre y la aproximación. Dentro de estos modelos, podemos encontrar los que aplican un razonamiento aproximado, como son la lógica difusa y los modelos probabilísticos, y la aproximación funcional y métodos de optimización, donde se engloban las metaheurísticas y las redes neuronales. A diferencia de los modelos de la Hard Computing, muchos de los modelos utilizados por la Soft Computing son estocásticos y capaces de trabajar con datos ambiguos y que, en ocasiones, contienen ruido. A la vez, mientras los métodos de Hard Computing son secuenciales, los de la Soft Computing pueden trabajar en paralelo y sus respuestas son aproximadas, requiriendo normalmente un tiempo de cómputo menor.

Centrándonos en los modelos de la Soft Computing, y más concretamente en los métodos de aproximación/optimización, éstos se dividen en dos partes diferenciadas. Por un lado, contamos con las metaheurísticas. Definimos heurística como un algoritmo que trata de encontrar soluciones viables, intercambiando precisión y calidad por esfuerzo computacional, o dicho de otro modo, eficiencia en tiempo y espacio. Las metaheurísticas aprovechan esta definición y la usan como la base para ser aplicadas a diferentes tipos de problemas de optimización realizando pequeños cambios para adaptarse a dicho problema [Glover 06, Talbi 09]. La diferencia entre ambas reside en que las metaheurísticas extienden las capacidades de las heurísticas, combinando procedimientos usando una estrategia de mayor nivel. Estos procedimientos pueden ser tan simples como una manipulación de la representación o tan complejos como la creación de una metaheurística completa.

Por otro lado, en la rama del razonamiento aproximado, encontramos la lógica difusa, donde se encuentran englobados, entre otros, los sistemas basados en reglas difusas. En este trabajo, dentro del amplio abanico de técnicas disponibles, nos centraremos en los algoritmos genéticos y la lógica difusa, siendo éstas la base del trabajo realizado, y de las que hablaremos con más detalle en las siguientes secciones de este capítulo.

2.1.1 Metaheurísticas

En las últimas décadas, las metaheurísticas han sido ampliamente utilizadas para resolver problemas complejos de optimización. Muchos de estos algoritmos se han inspirado en la naturaleza y han

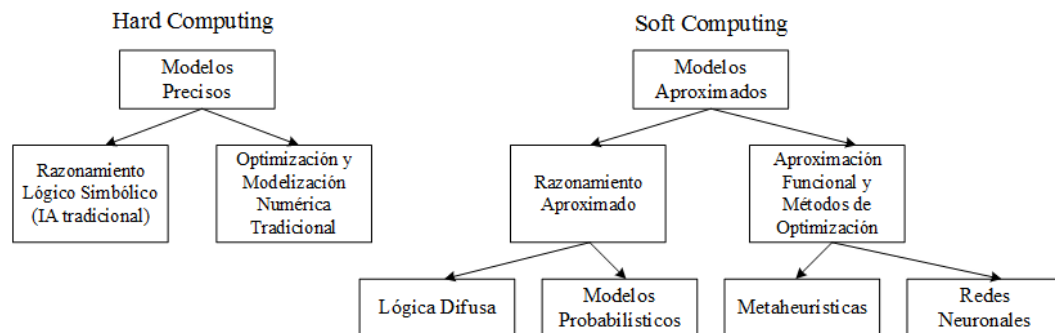


Figura 2.1: Clasificación de técnicas de Hard y Soft Computing

obtenido un gran rendimiento en problemas de alta dimensionalidad. Algunas de las características de este tipo de modelos expuestas por Blum y Roli en [Blum 03] son las siguientes:

- ◊ Las metaheurísticas son estrategias que guían el proceso de búsqueda. Hoy en día, las metaheurísticas más avanzadas mantienen alguna estructura de memoria en la que acumulan la información que han obtenido hasta el momento de cara a utilizarla en futuras iteraciones.
- ◊ Su objetivo es explorar el espacio de búsqueda de manera eficiente para así encontrar soluciones óptimas o muy cercanas al óptimo.
- ◊ El rango de técnicas englobadas bajo la definición de metaheurística va desde una simple búsqueda local hasta procesos complejos de aprendizaje.
- ◊ Estos algoritmos son aproximados y normalmente no determinísticos.
- ◊ Pueden incorporar mecanismos para evitar quedar atrapados en áreas determinadas del espacio de búsqueda, conocidas como óptimos locales.
- ◊ Las metaheurísticas no están definidas para un único problema, siendo posible su adaptación a problemas de diferentes ámbitos.
- ◊ Las metaheurísticas pueden hacer uso de conocimiento de un dominio específico en forma de heurísticas, controlando a éstas desde un nivel de estrategia superior.

Conociendo los muchos posibles tipos de clasificación de metaheurísticas, como los expuestos en [Crainic 03], [Brownlee 11] o [Glover 06], y en aras de la simplicidad y la concreción, queda fuera del ámbito de esta tesis hacer una enumeración exhaustiva de las diferentes categorías de metaheurísticas. Por ello, en este capítulo describiremos cuatro de las principales clases de metaheurísticas que podemos encontrar en la literatura, las cuáles se definen a continuación:

2. Estado del Arte

1. *Algoritmos basados en trayectoria*. Los algoritmos englobados en esta categoría parten de un punto inicial y van actualizando la solución actual a través de la exploración de su vecindario, creando de esta manera una trayectoria. La búsqueda se da por finalizada una vez se alcanza el número de iteraciones establecido, se encuentra una solución de calidad aceptable o se detecta un estancamiento. Dos de los algoritmos que se pueden denominar basados en trayectoria son Recocido Simulado, del inglés *Simulated Annealing* (SA) [Van Laarhoven 87] y la Búsqueda Tabú [Glover 86].
2. *Algoritmos evolutivos*. Estos algoritmos están inspirados por el proceso de los mecanismos de evolución biológica. Estos mecanismos describen cómo evolucionan los individuos a través de la modificación y propagación del material genético. En esta categoría entran, entre otros, los Algoritmos Genéticos (GA) [Holland 75] y Evolución Diferencial, del inglés *Differential Evolution* (DE) [Neri 10].
3. *Algoritmos basados en enjambres*. Basados en inteligencia colectiva, estos algoritmos usan la cooperación de un amplio número de agentes homogéneos en el entorno para resolver el problema en cuestión. Dicha inteligencia es descentralizada, auto-organizativa y distribuida a través del entorno. Ejemplos de este tipo de algoritmos son Algoritmo de Optimización de Partículas, del inglés *Particle Swarm Optimization* (PSO) [Kennedy 10] y Optimización de colonia de hormigas, del inglés *Ant Colony Optimization* (ACO) [Dorigo 97].
4. *Algoritmos probabilísticos*. Se pueden definir como métodos de optimización estocásticos que guían su búsqueda usando modelos probabilísticos contruidos a partir de las soluciones candidatas. Una parte amplia de este tipo de algoritmos son los denominados Algoritmos de Estimación de Distribuciones, del inglés *Estimation of Distribution Algorithms* (EDA), donde podemos encontrar métodos como la CE [Rubinstein 99] y CMA [Hansen 01].

Desde su formulación, muchos de estos algoritmos se han utilizado en problemas de optimización, como se verá más adelante. En las siguientes secciones, nos centramos en el desarrollo de los GAs así como de la CE, siendo estas dos técnicas las bases del método propuesto durante esta tesis. En secciones posteriores se justifica el por qué de la elección de dichas técnicas.

2.1.2 Algoritmos Genéticos

Los GA son metaheurísticas de búsqueda que imitan los procesos de selección natural. La idea principal a partir de la cual se define su funcionamiento es mantener una población de individuos,

cuyos cromosomas representan soluciones candidatas a un problema dado. Dichas soluciones evolucionan con el tiempo mediante un proceso de competición y variación controlada. Cada individuo de la población tiene asociado un valor que define la calidad de dicho individuo. Dicho valor es conocido como fitness y se calcula, normalmente, evaluando el individuo con la función (o funciones, en el caso de problemas multiobjetivo) objetivo. La calidad de dicho valor será el factor que determinará cuáles son los individuos utilizados para crear otros nuevos a partir de diferentes operadores utilizados.

En su versión más clásica, un GA mantiene una población de soluciones candidatas, o individuos. Cuatro procesos son aplicados iterativamente en dicha población, hasta cumplir un criterio de parada determinado que podría consistir, por ejemplo, en un número determinado de iteraciones:

1. *Operador de selección*: Este proceso selecciona dos (o más) individuos para ser padres, dependiendo de su fitness. Los individuos que son mejores que otros, es decir, cuyo valor fitness es el más alto, en caso de estar ante un problema en el que buscamos maximizar el valor retornado por la función de evaluación, o más bajo en caso contrario, tienen una mayor probabilidad de ser seleccionados por el operador. Las características de los padres seleccionados se transferirán a sus descendientes, o hijos, que son creados usando los otros operadores.
2. *Operador de cruce*: Este operador crea nuevos individuos combinando los padres. Un valor, conocido como probabilidad de cruce (P_{cruce}), está asociado a este tipo de operador. Dicho valor suele encontrarse en el rango $[0.5, 1.0]$ y determina el uso o no de este operador sobre un determinado par de individuos. Esto es, el operador sólo se aplica en el caso de que un número aleatorio supere el valor del parámetro.
3. *Operador de mutación*: Este proceso modifica un individuo aplicando algún tipo de cambio en su cromosoma. Como el operador de cruce, este operador también tiene un parámetro asociado, conocido como probabilidad de mutación ($P_{mutacion}$). En la mayoría de los casos, este parámetro oscila en el intervalo $[0.0, 0.3]$. Como en el caso de la probabilidad de cruce, el valor de la probabilidad de mutación determina su aplicación o no sobre el individuo en cuestión.
4. *Método de reemplazo*: Este proceso define cómo los nuevos individuos reemplazan, total o parcialmente, a los individuos en la población actual. Hay dos tipos de métodos de reemplazamiento. El primero, conocido como modelo generacional, reemplaza toda la población por

2. Estado del Arte

los hijos creados. El segundo, el modelo estacionario, sólo reemplaza una pequeña parte de la población.

Desde su aparición en los 70 de la mano de Holland [Holland 75], y gracias a su adaptabilidad a problemas difíciles, han aparecido en la literatura aplicados en solitario en trabajos como en [Osaba 13], donde un GA basado en islas, o poblaciones separadas, se aplica a problemas de rutas; o en [Osaba 14], donde varios tipos de GAs se usan para resolver problemas de optimización combinatoria. También se han combinado con diferentes tipos de técnicas para resolver otros tipos de problemas. Por ejemplo, en [Adeli 11] se presentan tres sistemas de diagnóstico de enfermedades usando reconocimiento de patrones basado en NN y GA. Por otro lado, se ha utilizado un GA para optimizar las reglas y funciones de pertenencia de dos controladores difusos utilizados para un modelo de control en intersecciones señalizadas presentado en [Qiao 11]. Otro ejemplo es el de [Bevilacqua 12], donde, para optimizar el diseño de redes de distribución, se utilizan GAs multiobjetivo.

Por la parte correspondiente a la lógica difusa, los GAs se han utilizado en muchos casos para mejorar o ‘afinar’ diferentes componentes de un FRBS. En [Onieva 12], un GA es utilizado para obtener la mejor base de reglas posible para un controlador que maneja un vehículo autónomo en intersecciones. En [Cordón 01c], se da una amplia explicación de GAs usados para optimizar diferentes partes de un FRBS. Los GAs y las estrategias evolutivas se utilizan de manera combinada en [Cordón 01b] para aprender y construir, a partir de ejemplos de bases de datos dadas, bases de reglas para sistemas difusos.

Centrándonos en la aplicación de los GAs a los ITS, podemos encontrar trabajos en la literatura como [Li 07], donde GA es utilizado para planear rutas de manera óptima. También podemos encontrar otro artículo donde se busca utilizar GA para encontrar un conjunto de escenarios dadas ciertas restricciones en [Gholamjafari 14]. Estos escenarios serán utilizados como conjunto de test en el campo de mejoras de los sistemas de frenado del vehículo ante una situación de peligro para el peatón. Siguiendo esta línea, en [Zou 13] se aplica un GA para modelar y calibrar los diferentes parámetros de estrategias de ITS para la reducción de la congestión y el mantenimiento de los niveles de servicio. En [Kanarachos 15], GA se utiliza para diseñar un sistema de suspensión inteligente con el objetivo de crear el actuador más pequeño posible.

La elección de GA como parte del algoritmo híbrido desarrollado se justifica con lo mencionado anteriormente: la gran capacidad de adaptación por parte de esta metaheurística poblacional la hace adecuada para tratar con ella sea cual sea el problema en el que se esté trabajando. Además, como

valor añadido, el fácil tratamiento de sus operadores, pudiendo cambiar estos dependiendo de la temática en cuestión, hace que se tenga una mayor variedad de posibles opciones a las que acudir.

Para cerrar esta sección, en el Algoritmo 1 se encuentra un ejemplo de la estructura que sigue un GA. En primer lugar, se inicializa la población de individuos (línea 2) y se evalúa dicha población (línea 3). Tras ello, y mientras no se alcance la condición de parada, se aplican los operadores mencionados anteriormente (líneas 5 a 8) y, posteriormente, se reemplaza la población actual por los nuevos individuos creados (línea 9). Cabe destacar que este algoritmo sigue la aplicación de un GA básico. Otras ejecuciones pueden cambiar el orden de los operadores [Osaba 13], o reemplazar parcialmente la población.

Datos: Pob_{tam} , P_{cruce} , $P_{mutacion}$

Resultado: *Mejor individuo encontrado*

```

1  $t \leftarrow 0$ 
2  $P_0 \leftarrow$  Inicializar Poblacion de  $Pob_{tam}$  individuos
3 Evaluar  $P_0$ 
4 mientras Condicion de parada no alcanzada hacer
5    $Padres \leftarrow$  Seleccionar padres de  $P_t$ 
6    $Hijos \leftarrow$  Cruce( $Padres, P_{cruce}$ )
7    $Hijos \leftarrow$  Mutate( $Hijos, P_{mutacion}$ )
8   Evaluar  $Hijos$ 
9    $P_{t+1} \leftarrow$  Reemplazamiento con la Población actual  $P_t$  e  $Hijos$ 
10   $t \leftarrow t + 1$ 
11 fin
```

Algoritmo 1: Pseudocódigo del proceso seguido por un GA.

2.1.3 Entropía Cruzada

El método de CE fue propuesto por Rubinstein en 1997 [Rubinstein 97]. Fue creado como un método adaptativo de estimación para eventos que ocurrieran con probabilidades muy bajas, también llamados *rare-event probabilities*, y para optimización combinatoria. El nombre de la técnica viene del método de entropía cruzada de Kullback-Leibler [Kullback 51] para medir la cantidad de información necesaria para identificar un evento a partir de un conjunto de probabilidades. CE tiene tres fases principales. las cuáles se ejecutan iterativamente:

1. Generación de un número $Ejemplos_{num}$ de ejemplos aleatorios siguiendo una distribución normal con media \bar{x} y desviación típica σ .

2. Estado del Arte

2. Seleccionar un número $Ejemplos_{seleccionados}$ de ejemplos con mejor valor fitness a partir del conjunto creado en la fase anterior.
3. Actualizar los valores de \bar{x} y σ de acuerdo con el valor fitness de los ejemplos escogidos.

Conforme el número de iteraciones va avanzando, \bar{x} converge hacia los puntos con mejores resultados mientras que σ disminuye hasta que ambos se centran en el área de mejores soluciones encontradas en el dominio. CE cuenta con un parámetro, $Learn_{rate}$, cuyo valor se encuentra normalmente en el intervalo $[0.7, 0.9]$ [De Boer 05]. Este parámetro se utiliza para actualizar los valores de media y desviación durante la ejecución del algoritmo con las medias y desviaciones de los nuevos ejemplos seleccionados. Cuanto mayor sea este valor, más se parecerá la media (y desviación) a la media de los ejemplos seleccionados y, por tanto, la convergencia hacia la solución será mucho más rápida.

La CE se puede aplicar en problemas de estimación u optimización [Rubinstein 99], y, además, puede ser usado en diferentes campos. Por ejemplo, en [Haber 10], CE modela un sistema de control difuso para un proceso de perforación. Por otro lado, en [Garcia-Villoria 10], un problema de programación combinatoria es resuelto usando un CE. En [Xia 12] se expone otro caso donde CE se utiliza para un problema de toma de decisiones, ayudando a determinar los pesos óptimos de los atributos. En [Ilanovsky 11] se trata un problema real que tiene como temática la producción de imágenes en tiempo real con vehículos no pilotados utilizando CE en la simulación de los escenarios. En medicina, podemos encontrar una modificación de la CE usada para analizar la relación entre el flujo y la presión de la sangre en [Zhang 09].

En el Algoritmo 2 se encuentra el proceso que sigue el método en pseudocódigo. Es importante tener en cuenta que el algoritmo está presentado para un problema unidimensional; en el caso de contar con más dimensiones, \bar{x} y σ deben ser vectores y cada una de dichas dimensiones debe ser tratada de manera independiente al resto.

Se ha escogido la CE como la otra mitad del algoritmo base de esta tesis por su capacidad de explotación del espacio de soluciones. Al centrarse cada vez más en la zona donde residen las soluciones más prometedoras, es la pareja perfecta para la exploración ofrecida por el GA, guiando CE, por tanto, la búsqueda. Por otro lado, al ofrecer también en cierta manera una población de individuos, no resultará difícil combinar las soluciones aportadas por ambos algoritmos.

Datos: $Ejemplos_{num}, Ejemplos_{actualizar}, Learn_{rate}$

Resultado: *Mejor individuo encontrado*

```

1  $\bar{x} \leftarrow$  Inicializar Medias
2  $\sigma \leftarrow$  Inicializar Desviaciones
3 mientras Condicion de parada no alcanzada hacer
4    $Ejemplos \leftarrow$  Generar  $Ejemplos_{num}$  Ejemplos con distribución normal  $N(\bar{x}, \sigma)$ 
5   Evaluar  $Ejemplos$ 
6    $Ejemplos_{seleccionados} \leftarrow$  Seleccionar los  $Ejemplos_{actualizar}$  mejores individuos de
      $Ejemplos$ 
7    $\bar{x} \leftarrow (1 - Learn_{rate}) \cdot \bar{x} + Learn_{rate} \cdot \text{Media}(Ejemplos_{seleccionados})$ 
8    $\sigma \leftarrow (1 - Learn_{rate}) \cdot \sigma + Learn_{rate} \cdot \text{Varianza}(Ejemplos_{seleccionados})$ 
9 fin
```

Algoritmo 2: Pseudocódigo del proceso seguido por la Entropía Cruzada

2.1.4 Hibridación de técnicas

A pesar de su éxito en muchos y variados problemas y el gran número de técnicas existentes, las metaheurísticas cuentan con varios problemas a los que no se les ha encontrado una solución por el momento. Uno de ellos es la variabilidad de su rendimiento cuando se escoge un mismo algoritmo para diferentes problemas, ya que dependen de las características de la función a optimizar. Otro, relacionado con el anterior, es la selección del algoritmo apropiado para cada tipo de problema. Centrando nuestra atención en el primer problema, uno de los enfoques tomados en la literatura para abordarlo es la combinación de dos o más algoritmos para obtener uno mejor o contrarrestar las deficiencias que presentan por separado. Esta combinación es llamada Hibridación [Topcuoglu 07].

La hibridación de técnicas es un tema importante en la literatura como queda constadado en [Purwar 15] y [Hernández 13]. Este concepto se basa en combinar dos o más técnicas para crear sinergia entre ellas y mejorar su rendimiento, además de disminuir o suplir sus carencias. Las técnicas utilizadas suelen obtener un buen rendimiento para resolver un problema específico, y su combinación puede hacer que se mejoren los resultados obtenidos si se comparan con la ejecución en solitario de cada una de sus partes.

Los algoritmos híbridos han demostrado ser prometedores en muchos campos en general. Algunos ejemplos son, entre otros, [Purwar 15], donde se combina la técnica de clúster K-means con un perceptrón multicapa para crear un modelo de predicción. Dicho modelo elige una técnica entre 11 propuestas para su utilización en datos médicos incompletos (missing values). [Sangsawang 14] propone dos enfoques híbridos, un GA con autoajuste adaptativo y un PSO con distribución Cauchy para un problema con restricciones y los compara con metaheurísticas clásicas. Por último, se puede

2. Estado del Arte

encontrar un survey reciente sobre la hibridación, ya no sólo de metaheurísticas entre sí, sino con otros tipos de técnicas como los algoritmos exactos, en [Blum 11].

Centrándonos en los ITS, en los últimos años se han aplicado varios algoritmos híbridos con diferentes finalidades. Por ejemplo, [Liu 15] presenta una combinación entre PSO y NN de cara a entrenar la estructura y los parámetros de la red neuronal y predecir con ella la velocidad de los vehículos en una carretera de Shanghai. En [Dib 15] se desarrolla un algoritmo híbrido combinando GA con un algoritmo de búsqueda local para un problema de asignación de rutas a vehículos. En la experimentación, se utilizan tanto instancias creadas aleatoriamente como aquellas basadas en rutas reales. Otro problema de asignación de rutas a vehículos, esta vez con ventana de tiempo, se intenta resolver en [Zheng 14] con una hibridación entre un ACO y un GA. En [Ning 15] se plantea la resolución de un problema de planificación para el tránsito rápido de autobuses combinando GA con un SA y un método de escalado. Se busca que el SA incremente la capacidad como algoritmo de búsqueda del GA mientras que el método de escalado aumenta la diversidad de la población. Se utilizan, para este caso, datos reales.

En el campo de la optimización en particular podemos encontrar también variedad de hibridaciones. Algunos ejemplos son los presentados en [Abd-El-Wahed 11], donde se combinan las técnicas PSO y GA para actualizar las partículas del primer método, usando dicha hibridación en funciones multimodales extraídas de la literatura. También contamos con [Hernández 13], en el cual se presenta un método que combina DE con un algoritmo Hill Climbing para optimización con restricciones. Por último, en [Mandal 11] se puede encontrar DE, en este caso combinado con un algoritmo de búsqueda local, que se aplica posteriormente a problemas de optimización del mundo real.

Por otra parte, surge la necesidad de una clasificación de métodos híbridos debido a cómo se ha extendido durante la última década la hibridación de diferentes algoritmos de manera que se cree una sinergia entre ellos, mayoritariamente para suplir las debilidades de uno de los métodos con mecanismos o características de otro, y viceversa. Ejemplos de estas debilidades son, por ejemplo, la falta de explotación del espacio de búsqueda por parte de metaheurísticas basadas en poblaciones [Talbi 02] o, en el caso de algoritmos basados en trayectorias, la facilidad con la que pueden quedarse atrapados en óptimos locales [Wang 04]. En el caso de algoritmos probabilísticos como DE, se han identificado problemas como la forma específica en la que se crean los nuevos individuos o el potencial de generar sólo un número limitado de soluciones en una generación [Segura 15].

Se pueden encontrar diferentes taxonomías en la literatura que buscan clasificar los algoritmos híbridos siguiendo una serie de pautas o guías. Por ejemplo, en [Talbi 02], se presenta una de ellas, además de un amplio número de hibridaciones clasificadas con esta taxonomía. El artículo presentado

en [Raidl 06] continúa con la idea desarrollada en el artículo anterior, combinando el punto de vista expuesto en [Cotta-Porras 98] y en [Alba 05] por Blum con la taxonomía de Talbi. La taxonomía más actual encontrada sobre este tema se encuentra en [Jourdan 09], donde se extiende, de nuevo y siguiendo un enfoque distinto, lo aportado en [Talbi 02] de manera que se tienen en cuenta esquemas cooperativos entre métodos exactos y metaheurísticas. Basándonos en las taxonomías anteriores, se escoge la propuesta en [Talbi 02] para realizar esta clasificación, dado que es la más utilizada incluso por los artículos más actuales.

Siguiendo, por tanto, esta taxonomía, podemos dividir las hibridaciones dependiendo del diseño y la implementación llevada a cabo. Para el caso del diseño encontramos hibridaciones:

- ◇ De nivel bajo, dependiendo de si una función de una de las metaheurísticas es sustituida por otra, o alto, si las diferentes metaheurísticas completas están contenidas en la hibridación.
- ◇ De relevo, donde el conjunto de metaheurísticas se aplican una detrás de otra, o de trabajo en equipo, donde se aplican agentes de cooperación paralela.
- ◇ Homogénea, donde todos los algoritmos utilizan la misma metaheurística, o heterogénea, donde todas las metaheurísticas utilizadas son diferentes.
- ◇ Global, si todos los algoritmos usan el espacio de búsqueda completo, o parcial, si cada uno de los algoritmos utiliza su propio espacio de búsqueda.
- ◇ Especialista, que combina algoritmos que resuelven diferentes problemas, o general, donde todos los algoritmos intentan resolver el mismo problema de optimización.

Para la parte de implementación, la hibridación puede ser:

- ◇ Específica, si sólo resuelve un conjunto pequeño de problemas, o de propósito general.
- ◇ Secuencial, donde los algoritmos trabajan uno por uno, o paralela, donde cada algoritmo está trabajando al mismo tiempo que el resto.

Por otra parte, si las metaheurísticas usadas trabajan de manera paralela, se encuentran en una de las siguientes tres categorías:

1. Estáticas, si el número de tareas y la localización del trabajo se generan durante el tiempo de compilación.

2. Estado del Arte

2. Dinámicas, si el número de tareas se determina durante la compilación pero la localización del trabajo se determina y/o cambia durante la ejecución.
3. Adaptativas, donde las tareas pueden ser creadas o borradas como una función.

El algoritmo que ha sido desarrollado durante esta tesis se basa en la combinación de las dos técnicas nombradas anteriormente, GA y CE. Se busca con su combinación paliar las debilidades que muestran las técnicas por separado y lograr, de esta manera, una técnica capaz de usar el espacio de búsqueda de una manera más óptima, de cara a encontrar soluciones ya no sólo en un tiempo menor, sino de una alta calidad. En el capítulo siguiente se describirá la técnica y se ampliarán los detalles al respecto.

2.1.5 Lógica Difusa

La lógica difusa, introducida por Zadeh en 1965 [Zadeh 65], permite procesar información imprecisa usando reglas del tipo SI-ENTONCES, o *IF-THEN*. Surge de la necesidad de controlar sistemas de los que no se tiene más que descripciones lingüísticas, incompletas e inexactas, basadas muchas veces en apreciaciones subjetivas de las variables de control [Zadeh 96]. La lógica difusa se basa en el concepto de conjunto difuso, donde la idea principal es que los elementos tengan un grado de pertenencia a un conjunto. Dicho grado de pertenencia se define en el intervalo $[0, 1]$, en vez de únicamente un valor $\{0, 1\}$.

El grado de pertenencia de una variable a un conjunto viene dado por su función de pertenencia (Membership Function, MF). Dicha función en un conjunto difuso es una generalización de una función característica en un conjunto clásico, y representa el grado con el que un elemento pertenece a un conjunto concreto. A la hora de determinar esta función, se eligen funciones sencillas. Contamos pues con diferentes tipos, siendo las más comunes la función triangular, función gamma, función S, Gaussiana, y trapezoidal, entre otras [Klir 95, Cordón 01a]. Centrándonos en las funciones triangulares y trapezoidales, siendo éstas últimas las utilizadas en el trabajo que nos ocupa, podemos decir que las funciones triangulares son aquellas que cuentan con tres puntos característicos, y, como indica su nombre, la función tiene forma de triángulo. La Ecuación 2.1 muestra la formulación matemática de la misma.

$$A = \begin{cases} (x - a)/(m - a) & \text{si } x \in (a, m] \\ (b - x)/(b - m) & \text{si } x \in (m, b) \\ 0 & \text{en cualquier otro caso} \end{cases} \quad (2.1)$$

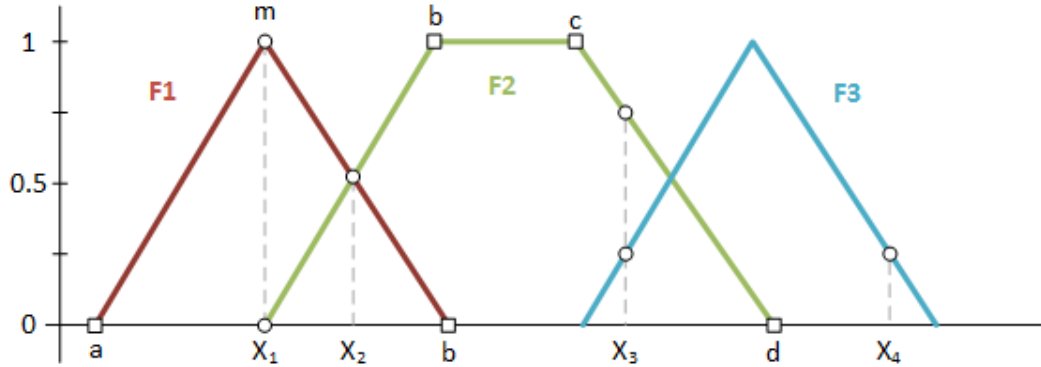


Figura 2.2: Ejemplo de funciones de pertenencia triangulares y trapezoidales

siendo a el punto donde comienza la función, m el punto donde la MF es igual a 1, y b el punto donde finaliza la función. Por otro lado, contamos con las funciones trapezoidales, las cuales cuentan con cuatro puntos. Se puede decir que estas funciones son una extensión de las triangulares ya que el punto extra con el que cuentan es el punto medio m que se transforma en un intervalo. La Ecuación 2.2 muestra la codificación de este tipo de funciones.

$$A = \begin{cases} (x - a)/(b - a) & \text{si } x \in (a, b] \\ 1 & \text{si } x \in (b, c) \\ (d - x)/(d - c) & \text{si } x \in (b, d) \\ 0 & \text{en cualquier otro caso} \end{cases} \quad (2.2)$$

donde a es el punto donde comienza la función, b y c son los puntos entre los cuáles la función de pertenencia toma valor 1, y d el punto donde ésta finaliza. Con la idea de ilustrar ambos tipos de funciones y dejar todo lo claro posible los puntos que las delimitan, un ejemplo gráfico se puede encontrar en la Figura 2.2, donde se representan varias funciones triangulares y una función trapezoidal.

Mientras que las funciones F1 y F3 representan funciones triangulares, la función F2 es trapezoidal. Contamos pues en dicha gráfica con cuatro elementos, definidos como $\{x_1, x_2, x_3, x_4\}$ con distintos grados de pertenencia a dichas funciones. Por ejemplo, el elemento x_1 tendrá un grado de pertenencia 1 a la función F1 y un grado de pertenencia 0 a las funciones F2 y F3. Además, es el punto a de la función F2. Por otro lado, x_2 cuenta con grado de pertenencia 0.5 a las funciones F1 y F2, mientras que x_3 tiene un grado de pertenencia 0.75 a la función F2 y 0.25 a la función F3. Por último, x_4 cuenta con un grado de pertenencia 0.25 a la función F3. De esta manera, se muestra como un elemento puede tener grados de pertenencia a dos o más clases diferentes (elementos x_1 ,

2. Estado del Arte

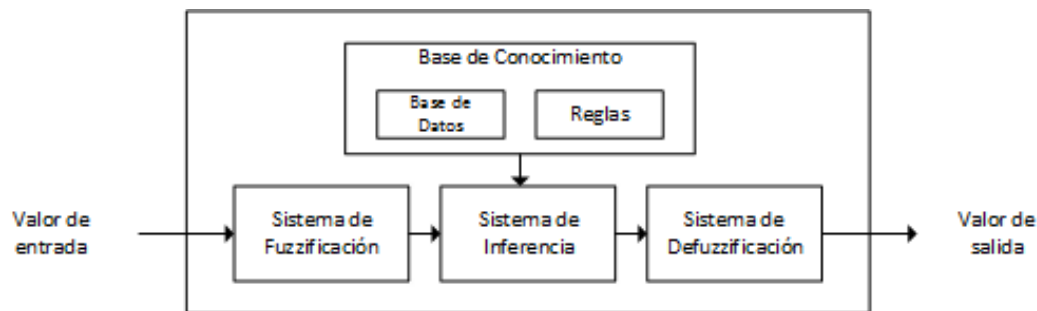


Figura 2.3: Ejemplo de un sistema basado en reglas difusas

x_2 y x_3), y como éste puede tener el mismo valor pero ser elementos completamente diferentes (elementos x_3 y x_4)

Para un ejemplo más práctico, por ejemplo en problemas de tráfico, se encuentra artículos como [Onieva 09], donde la lógica difusa se ha utilizado a menudo dado que permite que la información y las decisiones puedan ser tratadas con este tipo de reglas. Por ejemplo: *si la ocupación es alta y el flujo de coches es bajo, entonces existe congestión en la vía.*

2.1.5.1 Sistemas Basados en Reglas Difusas

Uno de los tipos de sistemas difusos más utilizados son los FRBS. Podemos definirlos como una extensión de los sistemas basados en reglas, dado que trabajan con reglas *IF-THEN* cuyos antecedentes y consecuentes están compuestos por reglas de lógica difusa, en vez de lógica clásica. Estos sistemas están formados por varias entradas, un sistema de inferencia difuso con un conjunto de reglas en su interior y una salida. Un ejemplo de esta clase de sistemas se encuentra en la Figura 2.3

Los FRBSs son utilizados en diferentes tipos de problemas de la vida real. En [Ghorbani 10], se utiliza uno de estos sistemas para simular diferentes políticas de gestión de energía en vehículos eléctricos. Clasificadores basados en reglas difusas y sistemas difusos son usados en [Iglesias 10] para reconocer acciones o actividades de los usuarios para predecir futuras acciones y monitorizar la salud de dichos usuarios remotamente. Podemos encontrar otro ejemplo en [Awan 11], donde se utiliza un FRBS para predecir el tiempo atmosférico.

En la literatura se distinguen, principalmente, dos tipos de FRBS en función del tipo de consecuente [Cordón 01a]:

1. FRBS de tipo Mamdani. Este tipo de sistemas utiliza variables de entrada y salida reales. Consta de un sistema de fuzzificación, que transforma dichas variables a su valor en los conjuntos difusos, un sistema de inferencia, la base de reglas, que contiene la información

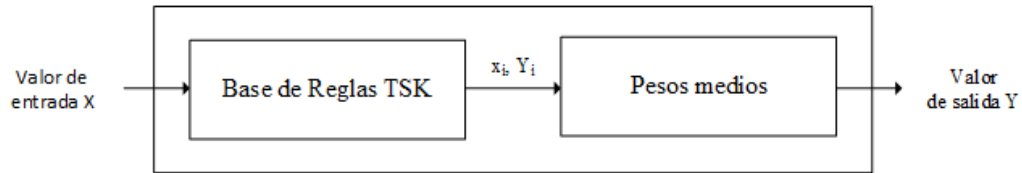


Figura 2.4: Ejemplo de un sistema de tipo TSK

sobre el problema en forma de reglas *IF-THEN*, y un sistema de defuzzificación, que vuelve a transformar los valores a su valor real. El resultado de la regla es, por tanto, un conjunto difuso. Un ejemplo sería "Si X_1 es ALTO y X_2 es MEDIO entonces Y es ALTO". La Figura 2.3 muestra un ejemplo de este tipo de sistema.

2. FRBS de tipo TSK (Takagi-Sugeno-Kang). En este caso, en vez de trabajar con reglas lingüísticas, se proponen reglas cuyo antecedente está compuesto por variables lingüísticas y el consecuente se representa como una función utilizando las variables de entrada. El resultado de la regla es, por tanto, una función. Siguiendo con el ejemplo anterior, "Si X_1 es ALTO y X_2 es MEDIO entonces Y es $f(X_1, X_2)$ ". De una manera más formal, la salida de un sistema TSK se obtiene mediante la media ponderada de las salidas individuales provistas por cada regla Y_i , y la entrada del sistema. La Figura 2.4 ilustra este proceso, a la vez que muestra la diferencia entre un sistema como el presentado en la Figura 2.3 y el actual, siendo dicha diferencia, como se ha dicho, la base de reglas utilizada y la inclusión de pesos medios para el cálculo de la salida.

Dentro de los FRBS, una de sus variantes es el llamado FRBS jerárquico (Hierarchical FRBS, HFRBS) [Fernández 09]. Esta clase de sistemas cuenta con varios FRBSs, los cuáles están unidos unos a otros de tal forma que la salida de uno de ellos es la entrada de otro.

Dependiendo de la forma en la que estructuren los sistemas [Benitez 13], contamos con tres modelos diferentes de HFRBS:

1. HFRBS en serie: la salida de un FRBS es la entrada del siguiente. También es posible incluir variables externas como una entrada del siguiente sistema.
2. HFRBS en paralelo: la estructura de los sistemas está organizada en capas. Las salidas de la primera capa son las entradas de los sistemas de la segunda capa, y así sucesivamente.
3. HFRBS híbridos: una combinación de los casos anteriores.

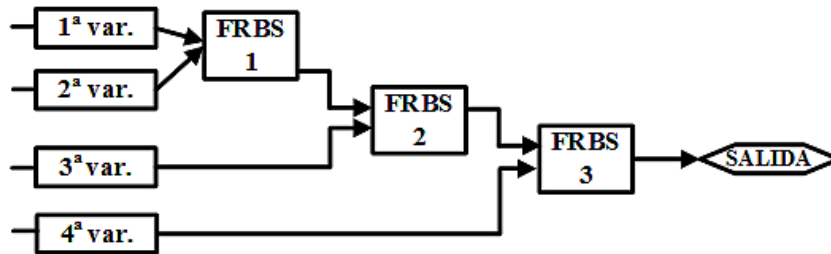


Figura 2.5: Tipos de jerarquías de FRBS: HFRBS en serie.

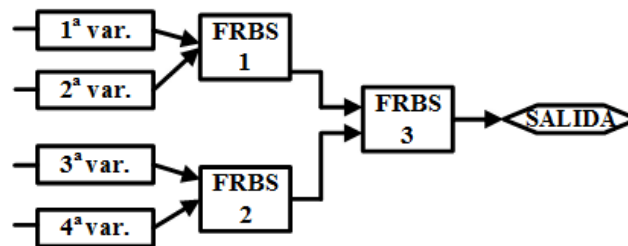


Figura 2.6: Tipos de jerarquías de FRBS: HFRBS en paralelo.

La Figura 2.5 muestra un ejemplo de un sistema de jerarquía en serie, mientras que la Figura 2.6 y la Figura 2.7 muestran sistemas paralelos e híbridos respectivamente.

Por otro lado, gracias a estas estructuras, los HFRBS son útiles para la mejora del equilibrio entre precisión e interpretabilidad en problemas con un gran número de variables, dando lugar a una posible solución al problema de ‘maldición de la dimensionalidad’ (curse of dimensionality problem), [Benitez 13].

Con el objetivo de reunir una lista de artículos sobre este tipo de sistemas para ampliar el conocimiento sobre estos y su aplicación, la Tabla 2.1 contiene las principales aportaciones de los HFRBS a la literatura en la última década. En dicha tabla se muestra el año de publicación, la

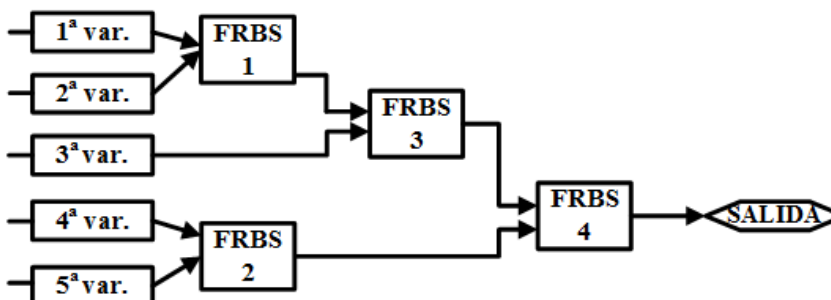


Figura 2.7: Tipos de jerarquías de FRBS: HFRBS híbrido.

2.2 Descripción de los Sistemas Inteligentes de Transporte

Año	Artículo	MF	Tipo de Sistema	Estructura	Aprendizaje
1995	[Shimajima 95]	Radial Basis	TSK	Híbrida	GA, Back-propagation
2004	[Chen 04]	Exponencial	TSK	Híbrida	Colonia Hormigas
2007	[Chen 07]	Exponencial	TSK	Híbrida, Serie	PIPE
2008	[Aja-Fernández 08]	Triangular	Mamdani	Paralela, Híbrida	FITM
2009	[Joo 09]	Triangular	TSK	Paralela	Greedy
2010	[Jelleli 10]	Gausiana	TSK	Paralela	Toma de Decisiones
2012	[Benitez 13]	Triangular	Mamdani	Serie	GA Multi-objetivo
2013	[Zhang 14]	Trapezio	TSK	Serie	GA Multi-objetivo
2016	[Lopez-Garcia 16a]	Trapezio	TSK	Paralela	Método Híbrido (GACE)

Tabla 2.1: Tabla de artículos en la literatura de Sistemas Jerarquicos Basados en Reglas.

referencia al artículo, las funciones de pertenencia utilizadas, el tipo de sistema, su estructura y el algoritmo de aprendizaje utilizado, detallado en cada uno de los artículos destacados. Conviene incurrir en la importancia del algoritmo de aprendizaje utilizado, dado que será éste el que determine, en gran medida, la evolución y calidad de las diferentes reglas y etiquetas de los sistemas, básicos para el rendimiento de los mismos.

Siendo esta únicamente una introducción, en secciones posteriores se hablará más detalladamente de la composición y aplicación de esta clase de sistemas al trabajo desarrollado y de cómo el algoritmo optimiza cada una de sus partes.

2.2 Descripción de los Sistemas Inteligentes de Transporte

Esta sección pretende dar cobertura a las diferentes definiciones existentes de qué son y para qué se utilizan los ITS. A su vez, se exponen los diferentes tipos de éstos, así como las diferentes áreas a los que son aplicados y las aplicaciones propiamente dichas que se realizan.

En primer lugar, se puede definir de manera sencilla los ITS como el vínculo entre las tecnologías de la información y la comunicación y los vehículos y redes que transportan personas o mercancías. Sin embargo, hay definiciones muy diversas realizadas por diferentes organismos e instituciones de gran relevancia. Por ejemplo, el Departamento de Transporte de EEUU lo define como:

Los ITS consisten en la aplicación de la tecnología avanzada de computación, electrónica y comunicaciones para incrementar la seguridad y la eficiencia del transporte de superficie.

Por otro lado, el Ministerio de Fomento Español lo define como:

Los ITS se pueden definir como un conjunto de aplicaciones avanzadas dentro de la tecnología informática, electrónica y de comunicaciones que, desde un punto de vista social, económico y

2. Estado del Arte

medioambiental, están destinadas a mejorar la movilidad, seguridad y productividad del transporte, optimizando la utilización de las infraestructuras existentes, aumentando la eficiencia del consumo de energía y mejorando la capacidad del sistema de transportes.

Los ITS surgen a finales de los 80 y principios de los 90 como alternativa sostenible al problema generado por la creciente demanda de movilidad, especialmente en ámbitos urbanos e interurbanos.

Las áreas donde se aplican estos sistemas son, principalmente, las siguientes:

- ◇ *Sistemas de Tratamiento del Tráfico Avanzado*: en esta categoría se integra el tratamiento de diferentes funciones de carretera. En este apartado se encontraría, por ejemplo, la predicción de las posibles retenciones de tráfico y las instrucciones que se darían a los vehículos de manera que se mejore la eficiencia de la red de la carretera. Una de las claves de este apartado sería la detección de los incidentes de forma que se redujera la congestión que provocan, y posibilitarían una actuación más rápida.
- ◇ *Sistemas de Información al Viajero*: Provee al viajero de información en sus vehículos, hogares o lugares de trabajo. Esta información incluye: localización de incidentes, información meteorológica, rutas óptimas, etc.
- ◇ *Sistemas de Control de Vehículo*: se busca en estos sistemas mejorar la seguridad durante el viaje y la eficiencia del vehículo. En este área se englobarían los sistemas de alerta de colisión, tanto automáticos (el coche activaría el freno automáticamente) como manuales (dependiendo del usuario), o la información y control de la infraestructura, como por ejemplo el manejo de la velocidad de los vehículos mientras van por una vía que tiene establecida una velocidad determinada.
- ◇ *Operaciones con Vehículos Comerciales*: los sistemas que se encuentran en esta categoría buscan mejorar la productividad y eficiencia de las flotas de camiones, furgonetas, y taxis.
- ◇ *Sistemas de Transporte Público*: en este punto se encuentran las soluciones que se aportan para mejorar la accesibilidad a la información de los usuarios del transporte público así como la mejora del horario que este mantiene y la utilización de las flotas de vehículos.
- ◇ *Sistemas de Transporte Rural*: esta opción resulta de las más complicadas para los ITS debido a las restricciones económicas relativas a las carreteras con poca densidad de vehículos en muchos lugares rurales.

2.2 Descripción de los Sistemas Inteligentes de Transporte

Por otro lado, los ITS intervienen en los diferentes tipos de transporte de maneras variadas:

- ◇ El transporte aéreo es uno de los tipos en los que más se ha avanzado en los últimos años, debido a la necesidad de ordenar el espacio aéreo con sistemas de localización, control de vuelo, o en las relaciones entre aviones y aeropuertos.
- ◇ En el caso del ferrocarril, siendo la seguridad uno de los objetivos más importantes, las circulaciones se han regulado utilizando los últimos avances tecnológicos, apoyándose en su singularidad de interacción continua entre los vehículos y la vía.
- ◇ Para transporte marítimo, los avances tecnológicos se han centrado en las telecomunicaciones, ya sea a la hora de navegar o de usar localización en un área determinada.
- ◇ Por último, el transporte por carretera es actualmente un foco de actuación constante que avanza a pasos agigantados dada la cantidad de problemas que presenta, siendo algunos de los más importantes la congestión en carretera, o la accidentalidad en la misma.

Centramos lo que resta de esta sección en el transporte por carretera, donde se encuentra englobada la parte principal del trabajo realizado en esta tesis. Como se ha dicho anteriormente, las razones impulsoras del desarrollo de los ITS en este campo son la accidentalidad, las congestiones de tráfico y el deterioro del medio ambiente, por lo que los avances se han centrado en temas como la seguridad, la capacidad de las infraestructuras y los efectos contaminantes de los vehículos. Podemos encontrar dos áreas:

1. **Infraestructuras inteligentes**, cuya finalidad es mejorar la seguridad del transporte público y privado desde el entorno de circulación, ya sea proporcionando instalaciones o servicios más eficientes. Un ejemplo podrían ser los sistemas de gestión de emergencias, de información meteorológica o información de tráfico y viaje.
2. **Vehículos Inteligentes**, mejorando la seguridad y movilidad de los vehículos instalando sistemas como sensores, equipos informáticos y de comunicaciones, que mitigarían las consecuencias de los accidentes que pudieran ocurrir. Los vehículos autónomos o los sistemas de asistencia a la conducción son áreas de interés dentro de este campo.

Un diagrama donde se resume la información de esta sección se encuentra en la Figura 2.8. En nuestro caso, donde se quiere predecir la congestión en una vía, y tal y como queda recalcado en dicho diagrama, el trabajo pertenece al área de infraestructuras inteligentes, y más concretamente, a los sistemas de tratamiento del tráfico.

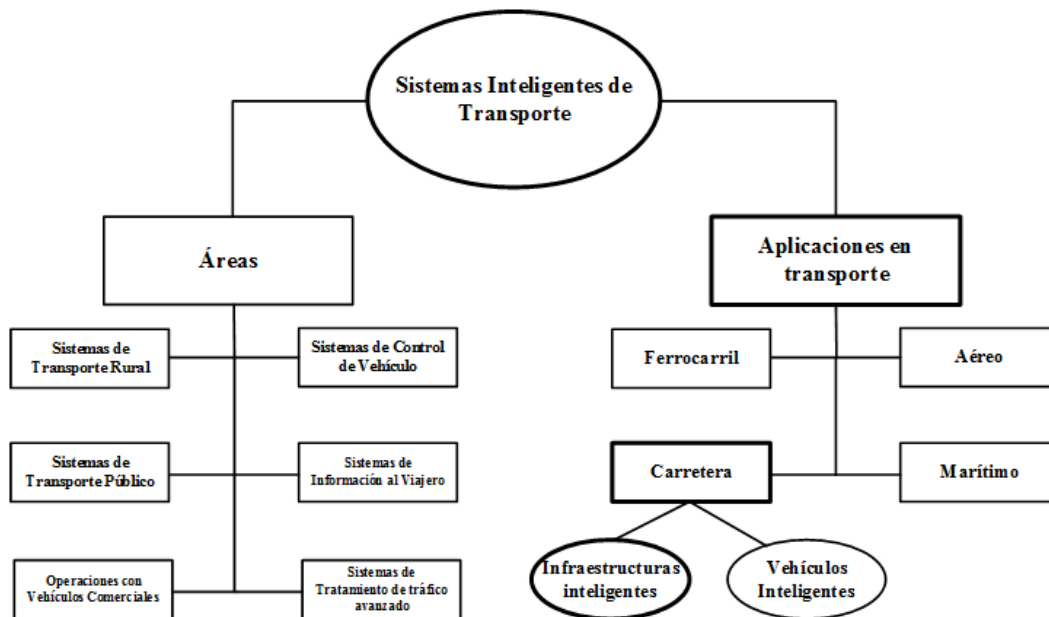


Figura 2.8: Diagrama resumen de los Sistemas Inteligentes de Transporte, sus áreas y sus aplicaciones

2.3 Optimización

A lo largo de esta sección, se describen los diferentes tipos de optimización que se pueden encontrar en la literatura en la Sección 2.3.1, un estudio de las diferentes técnicas de Soft Computing aplicadas a los diferentes tipos presentados en la Sección 2.3.2, y, por último, diferentes tipos de funciones benchmark que se usan en diferentes artículos, las funciones que contienen y sus diferentes características en la Sección 2.3.3.

2.3.1 Tipos de Optimización

En secciones anteriores se ha hablado brevemente de los tipos de optimización, siendo los más destacados la optimización numérica [Schwefel 81], lineal [Bertsimas 97], continua [Eiselt 87] o combinatoria [Papadimitriou 98].

Adentrándonos más en cada una de ellas, podemos definir la optimización numérica, también llamada solamente optimización, de la siguiente forma: Considerando una función $f(x)$, se busca un valor x por el cual la función f obtiene su valor máximo o mínimo y , tal que $Max/Min y = f(x)$.

Ambas variables x e y pueden ser tanto un único valor, como un vector de valores. La función f puede tener una serie de restricciones que se deben satisfacer, pudiendo diferenciar así optimización

con o sin restricciones. El ejemplo anterior es, por tanto, el caso más básico de optimización que nos podemos encontrar en la literatura [Nocedal 06].

En el caso de la optimización lineal, también llamada programación lineal, el objetivo general es minimizar una función objetivo lineal de variables reales continuas sujetas a restricciones de tipo lineal. Podemos contar con dos tipos de soluciones: una solución viable, es decir, aquella que satisface todas las restricciones, o una solución óptima, que a su vez se considera viable, siendo esta solución la que obtiene el menor valor aplicando la función objetivo. Para resolver este tipo de problemas podemos usar dos métodos: métodos simples [Lagarias 98], que buscan reducir este tipo de problemas a sistemas cuadráticos, de manera que puedan ser resueltos por valores únicos, o los llamados métodos *barrera* o de punto interior [Wright 05], derivados de la programación no lineal. Podemos encontrar un ejemplo de esta clase de optimización en [Murat 14], donde se busca minimizar el tiempo de acceso de los viajeros y el tiempo de viaje de las líneas de autobús de la ciudad de Izmir, en Turquía. Otro ejemplo lo podemos encontrar en [Bouras 13], donde se resuelve un problema lineal de un sólo objetivo con una combinación de técnicas.

En la optimización combinatoria, se tiene una familia de subconjuntos que forman parte de un conjunto finito, tales que el objetivo es encontrar uno de ellos que satisfaga la función objetivo. Ejemplos de este tipo de optimización pueden ser el problema del viajante de comercio (o *Traveling Salesman Problem* (TSP)) [Osaba 14, Zhang 12a], problemas de empaquetado [He 13, Silva 14] o el problema de la programación de producción discreta (o *Job Scheduling Problem* (JSP)) [Calis 15, Genova 15].

Centrándonos en la optimización continua, las variables toman valores dentro del dominio de los números reales [Wright 99]. Esta característica distingue este tipo de optimización de otras como la combinatoria o la discreta, donde las variables pueden ser binarias, enteros u objetos de un conjunto. Se han aplicado algoritmos iterativos a la optimización continua, los cuales generan una secuencia de valores en cada iteración para, finalmente, converger en una solución válida. Debido a su alta complejidad, se han aplicado diferentes técnicas a esta clase de problemas [Jourdan 09, Elsheikh 14, Holland 75], entre ellas, técnicas de inteligencia artificial.

Brevemente, se nombran a continuación otros tipos de optimización, dependiendo de diversos factores como:

- ◇ *Restricciones*: podemos contar con problemas sin restricciones o con restricciones. Los problemas con restricciones se pueden tratar como uno sin ellas si se sustituyen dichas restricciones por valores de penalización en las funciones objetivo [Jiang 08].

2. Estado del Arte

- ◇ *Objetivos*: la mayoría de los problemas de optimización tienen una única función objetivo. En el caso de contar con varias funciones objetivo, el problema se transforma en la búsqueda de valores que cumplan las restricciones propuestas, creando así soluciones viables [Coello 02].
- ◇ *Datos*: dependiendo del conocimiento o no de todos los datos que proporciona el problema, podemos contar con modelos determinísticos, donde los datos se conocen a ciencia cierta, o modelos estocásticos, que trabajan con la incertidumbre y, por tanto, con distribuciones de probabilidad [Casdagli 92].

2.3.2 Técnicas de Soft Computing aplicadas a Optimización

La literatura está llena de ejemplos de todos los tipos anteriormente mencionados, demostrando de esta manera que la optimización es un campo en continuo crecimiento y avance. Usando la base de artículos Scopus como referencia, a lo largo del año 2014 un total de 6139 artículos contienen uno de los tipos de optimización de los que se ha hablado en el apartado anterior entre las palabras clave asociadas al documento. Sobre esta cantidad, alrededor de un 80 % de los artículos contienen estudios sobre optimización numérica y lineal (40 % para cada uno, con 2471 y 2474 artículos respectivamente), mientras que la optimización combinatoria cuenta con un 11 % (678 artículos) y la optimización continua el restante 9 % (516 artículos). La Figura 2.9 resume todas estas cifras de manera visual.

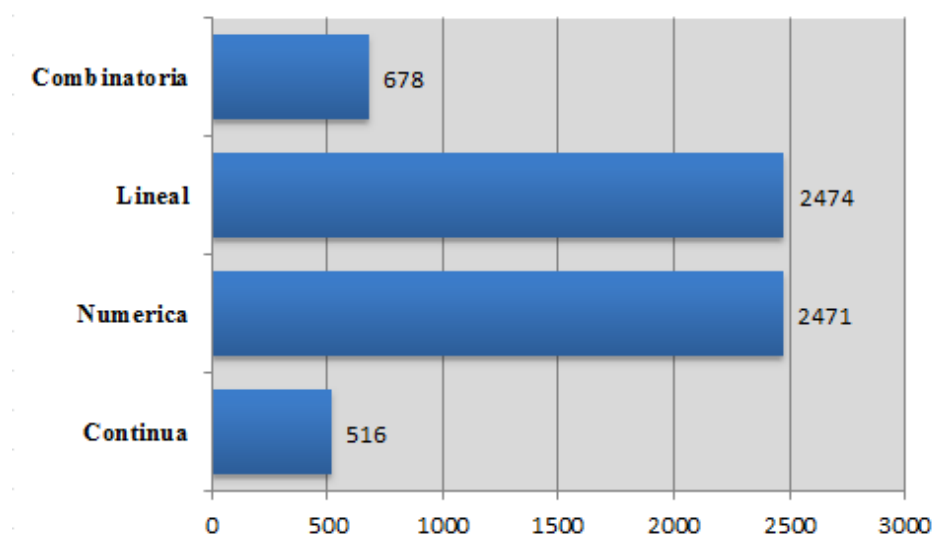


Figura 2.9: Artículos publicados durante el año 2014 en cada tipo de optimización (fuente: Scopus)

Podemos encontrar técnicas de Inteligencia Artificial, concretamente de Soft Computing, aplicadas a cada uno de los tipos de optimización. Para optimización lineal, podemos encontrar [Valizadeh 14], donde se aplica un modelo lineal para la cadena de suministro de biofuel en Malasia. Sobre dicho modelo se utiliza un PSO multi-objetivo y se compara con un algoritmo genético multiobjetivo del estado del arte llamado NSGA-II [Deb 02]. Otro ejemplo lo tenemos en [Azar 14] donde, entre todas las SVM que se utilizan para el diagnóstico del cancer de mama, se encuentra una basada en la programación lineal, la cual obtiene un más de un 97 % de acierto en el dataset utilizado. Por último, se presenta un algoritmo híbrido entre un GA y un PSO para resolver un problema lineal de dos niveles en [Kuo 15].

Como se ha mencionado en el apartado 2.3.1, hay ciertos problemas muy conocidos y utilizados en la literatura sobre optimización combinatoria. Uno de ellos es el TSP. La Soft Computing se ha utilizado mucho en este tipo de problemas, por ejemplo, en [Osaba 14], donde se aplica una metaheurística creada por el autor para resolver problemas de este tipo, o en [Okulewicz 13], donde se aplica un PSO a diferentes tipos de problemas de optimización combinatoria, entre ellos el TSP, pero además un problema de empaquetamiento (problema de la mochila).

Para optimización continua encontramos [Liao 14], donde se propone una ACO unificado para este tipo de optimización. Dicha propuesta combina tres variantes de este tipo presentadas en trabajos anteriores. La propuesta es aplicada a funciones benchmark utilizadas en diferentes congresos y revistas. Un algoritmo ACO se aplica también para la resolución de problemas de optimización continua en [Chen 14]. En [Toledo 14] se utiliza un GA con estructura de población jerárquica para resolver problemas continuos sin restricciones. Tanto este como un SA, DE y un PSO se utilizan en funciones benchmark en dicho artículo.

2.3.3 Funciones Benchmark

En el apartado anterior, algunos de los artículos mencionados han utilizado funciones benchmark extraídas o bien de revistas que las proporcionan, de congresos que cuentan con workshops dedicados a la temática de la optimización, o simplemente de páginas personales donde investigadores ofrecen dichas funciones para que cualquiera interesado en ellas pueda descargarlas. Esta sección se encarga de recoger algunas de esas recopilaciones de funciones, explicar sus características y para qué tipo de optimización se podrían utilizar dichas funciones.

En primer lugar, podemos definir benchmarking como la acción de comparar un proceso y su rendimiento con las técnicas más punteras en la temática en cuestión. Dicho rendimiento suele medirse con métricas como la calidad de la solución, el tiempo de cómputo de la misma y el coste, en

2. Estado del Arte

el caso de que tuviera [Lema 95, Hansen 09]. Por tanto, una función benchmark es aquella utilizada para medir el rendimiento de un algoritmo de cara a realizar una comparación justa con diferentes técnicas destacables en la literatura.

Según el teorema *no free lunch* enunciado en [Wolpert 97], si se comparan dos algoritmos con todas las posibles funciones de un conjunto, el rendimiento de ambos algoritmos sería, de media, el mismo. Por lo tanto, tener un conjunto de funciones extremadamente grande, donde se recoja todas las posibles es, en esencia, algo infructuoso. Por ello, cuando se evalúa un algoritmo, se debe buscar en qué tipo de problemas obtiene mejor rendimiento. Para esta tarea, debemos hacer un estudio de pocas pero bien elegidas funciones de diferentes tipos [Whitley 95]. Así, podríamos obtener conclusiones del rendimiento del algoritmo según el tipo de función en el que se aplica. Una función benchmark es, por tanto, una de las funciones que forman el conjunto a evaluar. La literatura cuenta con varios ejemplos de conjuntos de funciones, como pueden ser las extraídas de la tesis de De Jong [De Jong 75] o las que se pueden encontrar en el artículo [Eiben 97]. Por otro lado, la revista *Soft Computing* publicó un número especial dedicado a problemas de optimización continua de gran escalabilidad donde se recogían un total de 19 funciones [Lozano 11]. Entre estas funciones, se cuenta con las expuestas en el congreso dedicado a la computación evolutiva IEEE Congress on Evolutionary Computation (CEC) en una competición sobre este tipo de problemas en el año 2005. En el último año de esta competición (en el momento en el que se escribe este documento, 2015), cuenta con un total de 15 funciones de 1000 dimensiones cada una. Tanto en esta conferencia, como en la célebre Genetic and Evolutionary Computation Conference (GECCO), se realizan competiciones de optimización utilizando diferentes conjuntos de funciones, dependiendo de la competición, cuyos resultados y contenido son totalmente accesibles para aquellos que quieran utilizarlas. El algoritmo presentado en esta tesis participó en la Black-Box Optimization Benchmarking (BBOB) celebrada en la conferencia GECCO'15. En esta competición, se proponían un total de 24 funciones de cinco tipos diferentes¹: separables, con condicionamiento bajo o moderado, con condicionamiento elevado y unimodal, multimodales con estructura global adecuada, y multimodales con estructura global débil. Los resultados obtenidos con las funciones utilizadas en esta competición se mostrarán en capítulos siguientes.

Por último, diferentes páginas ofrecen benchmark para los diferentes tipos de optimización explicados en los apartados anteriores. Un ejemplo puede ser el creado por Hans Mittelman en ², donde se pueden encontrar actualizados diferentes benchmark a elegir.

¹ <http://coco.gforge.inria.fr/doku.php?id=bbob-2015>

² <http://plato.la.asu.edu/bench.html>

Nombre	Nº de Funciones	Tipos de Funciones	Referencia
De Jong	5	-	[De Jong 75]
Eiben	7	Unimodales, Multimodales, Separables	[Eiben 97]
SOCO	19	Escalables	[Lozano 11]
CEC	15	4 (ver Referencia)	Referencia ³
BBOB	24	5 (ver Referencia)	Referencia ⁴
Mittelmann	-	-	Referencia ⁵

Tabla 2.2: Tipos de benchmark encontrados en la literatura y sus características

Con la intención de resumir todo lo recogido en esta sección, en la Tabla 2.2 se encuentran las diferentes opciones propuestas.

³ <http://staff.ustc.edu.cn/~ketang/lsgo2015.html>

⁴ <http://coco.gforge.inria.fr/doku.php?id=bbob-2015>

⁵ <http://plato.la.asu.edu/bench.html>

No importa lo rápido que viaje la luz, siempre se encuentra con que la oscuridad ha llegado antes y la está esperando.

Terry Pratchett

3

Descripción del algoritmo

En este capítulo se tratarán todos los aspectos relacionados con el algoritmo que representa la base de este trabajo. La explicación y funcionamiento del método se encuentra en la Sección 3.1. Por otro lado, su aportación a los algoritmos híbridos, así como el área en el que se encuentra ubicado se muestra en la Sección 3.2. En la Sección 3.3 se habla de la aplicación del método a la optimización de los FRBS, mientras que en la Sección 3.4 se explica su aportación al campo de la optimización de funciones. Por último, en la Sección 3.5 se exponen algunos de los detalles que hay que tener en cuenta a la hora de aplicar el algoritmo.

3.1 GACE: Descripción y Funcionamiento

A lo largo de la última década, se han desarrollado una gran cantidad de metaheurísticas como se ha expuesto en el estado del arte. Sin embargo, pese a su éxito en las diferentes aplicaciones expuestas en el capítulo anterior, todavía sufren de varios problemas que se consideran aún abiertos, ya sea a la hora de elegir qué técnica es mejor para qué problema, o las debilidades que muestran cada una de ellas. Centrándonos en las últimas, y poniendo como ejemplo metaheurísticas poblacionales como GA y ACO, cuentan con problemas de explotación del espacio de búsqueda [Talbi 02], esto es, aunque ofrezcan una búsqueda de calidad, no consiguen alcanzar individuos de menor (mayor si es un problema de maximización) fitness o centrarse en el lugar concreto donde se encuentran dichas soluciones. Otro problema, teniendo en cuenta en este caso los algoritmos basados en trayectoria,

3. Descripción del algoritmo

como pueden ser el SA y la Búsqueda Tabú, es la posibilidad de quedar estancados en un óptimo local debido a su mal comportamiento en lo que a exploración del espacio de búsqueda se refiere [Wang 04]. Por último, otro problema, llamado el problema de Selección de Algoritmo, retrata el hecho de que, al contar con tantos algoritmos disponibles, no sea una tarea sencilla saber cuál obtendrá mejor resultado con la información disponible [Muñoz 13].

El algoritmo GACE, llamado de esta manera por combinar un Algoritmo Genético y un método de Entropía Cruzada (Genetic Algorithm with Cross Entropy en inglés) está diseñado con la idea de aprovechar la habilidad de exploración de un GA y la habilidad de explotación de CE en un problema de optimización dado. El hecho de usar estos algoritmos hace que se intente evitar el estancamiento por parte de CE en un óptimo local gracias a la habilidad de exploración del GA, mientras que se explota todo lo posible el espacio de búsqueda por la aplicación del CE, evitando de esta manera el problema presentado en el párrafo anterior sobre los algoritmos poblacionales. Por otro lado, el método se ha desarrollado además con la intención de crear una técnica capaz de adaptarse a los problemas de optimización que se le planteen con una carga mínima en lo que a cambio de estructura se refiere, ofreciendo a la vez un mejor rendimiento que las técnicas que la forman por separado. Dicho algoritmo se aplica a dos problemas, siendo el principal, y en el que se centra el desarrollo de la parte de experimentación de la tesis, la optimización de jerarquías de sistemas difusos con el objetivo de optimizar las entradas, etiquetas y reglas de cada uno de los sistemas que las forman. Para demostrar su capacidad de adaptación así como probar su rendimiento, se ha utilizado un segundo problema, siendo éste la optimización de funciones continuas, cuya configuración y resultados se expondrán en secciones posteriores. La técnica híbrida presentada en esta tesis representa una novedad en la temática de metaheurísticas híbridas ya que, hasta la fecha, y hasta el punto de conocimiento adquirido por su autor, no se había desarrollado dicha combinación en la literatura.

Conociendo los motivos de desarrollo de este método híbrido, se expone de manera detallada su funcionamiento general, que no variará sea cual sea el problema al que se aplique. En primer lugar, se inicializará la población de soluciones de forma completamente aleatoria. Dependiendo del problema que se esté tratando, esta inicialización se realizará de manera distinta, debido a la codificación de los individuos escogida para dicho problema. Una vez realizado este paso, la población se divide en dos subpoblaciones:

1. Población dedicada al GA, que contará con individuos a los que se les aplicarán los diferentes operadores de un GA. Estos individuos serán elegidos por el respectivo operador de selección escogido.

3.2 Combinación de las técnicas: Clasificación y aportación

2. Población dedicada al CE, con un número de individuos que se utilizarán en una CE. Los individuos que forman esta subpoblación son los mejores individuos existentes en la población actual de soluciones.

Los tamaños de subpoblación son elegidos por el usuario y su suma debe ser igual al tamaño de la población actual.

Una vez que ambas poblaciones se han escogido, cada una se usa de manera diferente:

- ◊ En la población dedicada al GA se aplican los operadores de cruce y mutación para generar los nuevos individuos. Los operadores concretos a utilizar variarán según el tipo de problema abordado en esta tesis y se especifican en la Sección 4.1.3.1.
- ◊ En el caso de la subpoblación dedicada al CE, se aplica el método CE a sus individuos. Primero se actualizan las medias y desviaciones empleando el Algoritmo 2 de la Sección 2.1.3 (líneas 7 y 8). Tras esto, los individuos son generados aleatoriamente siguiendo una distribución normal usando dicha media y desviación actualizadas. En la Sección 4.1.3.2 se encuentra este proceso aplicado, en este caso, al problema de predicción de congestión y sus individuos.

Tras aplicar ambos algoritmos a sus respectivas subpoblaciones, con los individuos resultantes de éstas se crea una población completamente nueva que contendrá, por tanto, individuos de los operadores del GA e individuos del método CE. Esta nueva población reemplaza en su totalidad a la anterior, por lo que GACE es un algoritmo generacional. Es interesante mencionar que se aplica elitismo en la nueva población, es decir, se mantiene en la población al mejor individuo encontrado en la anterior generación. La Figura 3.1 muestra el proceso descrito en esta sección.

Cabe destacar que los individuos pueden formar parte de ambas poblaciones a la vez. De esta manera se consigue que sean los mejores individuos los que vayan 'mejorando' con el paso de las generaciones, además de terminar centrando la mejora en un espacio de búsqueda concreto a lo largo de la ejecución del algoritmo.

3.2 Combinación de las técnicas: Clasificación y aportación

Tras haber realizado una explicación del funcionamiento del algoritmo, y habiendo realizado anteriormente una explicación detallada de la aplicación y uso de los Algoritmos Genéticos y la Entropía Cruzada en los diferentes temas de los que trata esta tesis, surge la necesidad de especificar las diferentes categorías, siempre siguiendo una taxonomía detallada, en las que residiría el algoritmo

3. Descripción del algoritmo

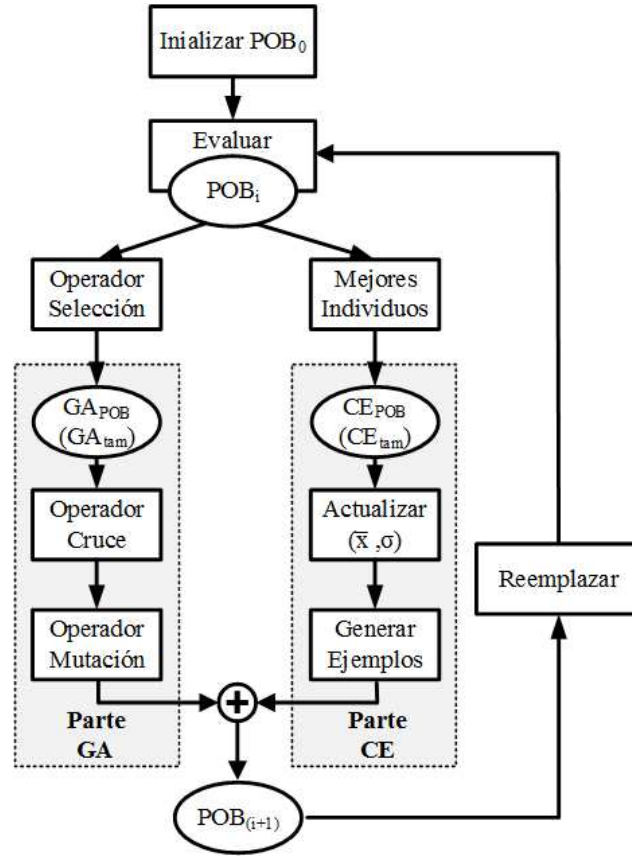


Figura 3.1: Pasos que sigue el algoritmo GACE

desarrollado según las técnicas encontradas en el estado del arte. Esta taxonomía es la aportada por Talbi en [Talbi 02], expuesta en la Sección 2.1.4.

En dicha taxonomía, y centrándonos en las características del diseño, el algoritmo propuesto recae en las siguientes categorías:

- ◊ Nivel Alto: la propuesta se encuentra dentro de las hibridaciones de nivel alto, ya que tanto GA como CE están contenidos en su totalidad en la hibridación y se mantienen todos los operadores de cada uno de los métodos hibridados.
- ◊ Trabajo en equipo: durante la ejecución del método propuesto, ambas partes trabajan en paralelo, uniendo los resultados de su ejecución al final de la generación en la creación de la nueva población.
- ◊ Heterogéneo: si tomamos CE por su naturaleza estadística, la hibridación es heterogénea, ya que GA es una técnica basada en población, siendo, por tanto, ambas diferentes. En cambio, si

3.3 Sistemas Basados en Reglas Difusas: Aplicación

pensamos en CE como un algoritmo poblacional, ya que mantiene una serie de individuos a lo largo de las generaciones, entonces estaríamos ante un caso de hibridación homogénea.

- ◇ Parcial: los algoritmos que forman la hibridación son aplicados a dos sub-poblaciones diferentes. A su vez, estas sub-poblaciones se forman de manera distinta: La sub-población con la que trabaja el GA se forma a partir del operador de selección mientras que con la que trabaja CE está formado por los mejores individuos de la población actual.
- ◇ General: el algoritmo se aplica a un problema dado con una función objetivo a optimizar, por lo que ambas partes funcionan de manera que se mejore los resultados obtenidos para dicha función, aplicándose, por tanto, al mismo problema objetivo.

Por otro lado, y teniendo en cuenta la parte de implementación, la hibridación desarrollada formaría parte de las siguientes categorías:

- ◇ Propósito general: aunque sí que es cierto que en este trabajo sólo se aplica la técnica desarrollada a ámbitos específicos como la predicción y la optimización de funciones, el algoritmo está creado de manera que pueda adaptarse a cualquier clase de problema. Bastaría con adecuar la inicialización y los métodos de cruce y mutación del GA, y el cálculo de la media y desviación para la actualización de los individuos en CE, al problema con el que se quiera tratar, además de, obviamente, la codificación de los individuos de la población de la que dependen ambas técnicas.
- ◇ Paralela: la aplicación de las diferentes partes de la técnica se realiza de manera separada gracias a las dos subpoblaciones que se mantienen, reduciendo así el tiempo de cómputo.

El objetivo tras la hibridación que se realiza en este trabajo, como ya se ha mencionado varias veces, es la aportación que ambas técnicas pueden hacer a sus diferentes ejecuciones. Por lo tanto, lo que se busca es el buen rendimiento de ambos algoritmos en problemas de optimización de índole totalmente diferentes. Será básico, como se demostrará en la experimentación, encontrar los parámetros correctos para su aplicación en cada uno de los diferentes problemas.

3.3 Sistemas Basados en Reglas Difusas: Aplicación

En la Sección 2.1.5 se ha hablado de los Sistemas Basados en Reglas Difusas y algunas de sus aplicaciones en problemas de la vida real. En la parte principal del trabajo desarrollado en esta tesis, se han utilizado este tipo de sistemas como herramienta para la predicción de congestión a corto

3. Descripción del algoritmo

plazo en diferentes puntos de una carretera. Concrétamente, este trabajo está enfocado en el tipo de sistemas HFRBS en paralelo (*Parallel HFRBS*, PHFRBS). Se ha elegido este tipo de jerarquía porque, de esta manera, todas las variables que pasan al sistema son procesadas al mismo tiempo y con la misma relevancia. Esto se puede observar comparando las Figuras 2.5 y 2.6. En el primer caso la cuarta variable entra al último sistema, influenciando a la salida sólo en ese momento. En el segundo caso, donde los sistemas están en paralelo, las cuatro variables se tienen en cuenta desde el primer momento, y son las salidas de sus sistemas (FRBS 1 y FRBS 2) las que, usadas como entradas en el último sistema, darán lugar a la salida de la jerarquía, teniendo, por tanto, la misma relevancia a la hora de calcular el resultado final.

Utilizamos esta jerarquía con una restricción: cada sistema sólo cuenta con dos entradas. Por lo tanto, si hay un número impar de variables, la última variable será la primera entrada del último sistema. Esto hace que las etiquetas y las reglas utilizadas para cada sistema no se dispare, lo que haría imposible en términos de tiempo de cómputo el funcionamiento del sistema, y en términos de representabilidad, su interpretación posterior.

Pongamos como ejemplo de esto lo siguiente: contamos con cuatro variables de entrada y tres etiquetas para cada una de ellas. Con un sistema no jerárquico, contamos con un total de $3^4 = 81$ reglas para cubrir cualquier combinación posible de éstas. En el caso de sistemas jerárquicos como los que nos ocupan, donde cada sistema cuenta con dos entradas, para cuatro entradas, contamos con $Etiquetas^{Entradas} \cdot Sistemas$ reglas, donde $Sistemas = Entradas - 1$, es decir, $3^2 \cdot 3 = 27$ reglas para cubrir todas las combinaciones. Generalizando, contamos con un total de $Etiquetas^N \cdot N - 1$ reglas para N entradas, sabiendo que se necesitan $N - 1$ FRBSs simples para tratarlas, por lo que el número de reglas crece exponencialmente. Poniendo como ejemplo los conjuntos de datos que se utilizarán para predicción de congestión, los cuáles constan de un total de 47 entradas y 3 etiquetas por entrada, si no se usara jerarquía se necesitarían un total de 3^{47} reglas, mientras que usando las jerarquías propuestas, contamos con un total de $3^2 \cdot 46 = 414$ reglas.

En el trabajo que nos ocupa, se han utilizado FRBS de tipo TSK, nombrados anteriormente, dado que permite un cálculo rápido de la salida del sistema. Se han utilizado trapecios para la codificación de las entradas y singletons (constantes) para las salidas. Para la inferencia se ha utilizado una T-norma mínimo y agregación por centro de masas (media ponderada).

El algoritmo propuesto realiza en estos sistemas la optimización de sus diferentes partes. De manera breve, dado que en futuras secciones se abordará con más detalle esta explicación, el algoritmo se encarga de optimizar, mediante la hibridización propuesta, las entradas de cada sistema, las etiquetas por las que se regirá la selección de las reglas, y las reglas en sí mismas.

3.4 Optimización de funciones: Aplicación

La principal idea tras la aplicación del algoritmo desarrollado en esta tesis a funciones benchmark es demostrar lo siguiente:

- ◇ Su capacidad de adaptación a otro tipo de problemas.
- ◇ Comprobar el rendimiento del algoritmo híbrido cuando es aplicado a funciones de optimización de diferente tipo y complejidad.
- ◇ La capacidad del algoritmo de superar, con la combinación de sus partes, a algoritmos en el estado del arte en esta materia.
- ◇ Comprobar si el cambio de ciertas partes del algoritmo, como pueden ser los métodos de cruce y mutación de la parte genética del mismo, afecta positiva o negativamente a su rendimiento en otros problemas.

Además de cada uno de estos puntos, es crucial para lograr el mayor rendimiento posible, conocer los valores correctos de los parámetros de cada una de las variables que influyen en cada generación del algoritmo. Como se verá en siguientes secciones, el estudio de dichos parámetros se ha llevado a cabo en la parte de la experimentación dedicada a estas funciones, centrándonos en las probabilidades de cruce y mutación por parte de la parte genética, y del número de individuos que se escogen para actualizar los valores de media y desviación típica que se utilizan en cada una de las ejecuciones de la Entropía Cruzada.

La razón por la que este estudio no se realiza antes de la aplicación del algoritmo a la optimización de los sistemas es simple: en el caso de la optimización de los sistemas, la experimentación está centrada en un ámbito general de estos parámetros, es decir, se toman los establecidos por defecto en la literatura (alta probabilidad de cruce, baja probabilidad de mutación, usar todos los individuos y probabilidad de actualización alta) y se centran los esfuerzos en la cantidad de individuos de cada subpoblación al igual que en la calidad de la solución, número de reglas y etiquetas, y reducción del tiempo de cómputo por la gran cantidad de datos escogidos. Posteriormente, y dada la respuesta positiva en cuanto a rendimiento dada por el algoritmo en este problema, se realiza dicho estudio para confirmar tal rendimiento en otro tipo de problemas y establecer estos parámetros sin estudios posteriores en otros ámbitos.

3.5 Aspectos a tener en cuenta en la codificación del algoritmo

En este apartado, se recalcan los aspectos a tener en cuenta encontrados durante la codificación y puesta a punto del algoritmo. Dichos aspectos no empeoran la ejecución del mismo, sino que traen en consecuencia tener en cuenta más detalles a la hora de adaptarlo a los diferentes problemas propuestos.

Precisamente, la necesidad de adaptar el algoritmo, concretamente la inicialización de la población y el cálculo de medias y desviaciones en la parte CE es una de sus limitaciones. Aunque con fácil solución, requiere el estudio previo del problema en cuestión, de manera que la inicialización de los individuos sea la correcta. En cada uno de los problemas que tratamos, la inicialización ha sido diferente, adaptándose al problema en cuestión.

Por otro lado, de entre todos los parámetros a tener en cuenta, el tamaño de la población principal, de la que dependen a su vez el tamaño de las subpoblaciones, merece una mención en esta sección. Aunque se realiza un estudio en capítulos posteriores, es importante conocer que un número elevado de individuos aumenta el tiempo de cómputo del algoritmo a la vez que no mejorará significativamente sus resultados. Esto hace que este tamaño sea pequeño, en torno a 50 individuos, y que ambas poblaciones tengan un máximo de este tamaño como máximo en sus ejecuciones *puras* (sólo GA o sólo CE). Esto deriva en una posible pérdida de diversidad. Al contar con tan poco número de individuos, si el número de generaciones es elevado, o incluso cuando no, todos los individuos de la población terminan siendo el mismo, provocando que, pese a contar con más 'tiempo' para mejorar, no se produzcan cambios. Las soluciones ofrecidas para solucionar esto han sido variadas. Finalizar la ejecución antes de cumplir el criterio de parada si la mejor solución no cambia en un porcentaje de generaciones dado, aumentar la probabilidad de mutación en el caso de que la diversidad disminuya, y el estudio de los parámetros de actualización en el caso del CE son algunas de ellas.

3.6 Implementación del algoritmo propuesto

La motivación de esta sección es mostrar las diferentes implementaciones que se han realizado del algoritmo adaptándose a los problemas propuestos. Recordando lo descrito en la Sección 3.1, y siguiendo el pseudocódigo descrito en el Algoritmo 3, correspondiente a la primera implementación publicada en [Lopez-Garcia 16a] y aplicada a problemas de predicción de congestión de tráfico, el método propuesto trabaja sobre una población de individuos, denominada Pob_t , que se inicializa de una forma diferente dependiendo de la codificación dada al problema que nos ocupe (línea 2),

aunque siempre de manera aleatoria. Tras dicha inicialización, la población actual se divide en dos subpoblaciones: una población dedicada al GA, llamada GA_{pob} , con un número de individuos GA_{tam} , y una población dedicada al CE, llamada CE_{pob} , con CE_{tam} individuos. Los tamaños de las subpoblaciones son escogidos por el usuario y su suma delimita el tamaño de la población actual. Es decir, $Pob_{tam} = GA_{tam} + CE_{tam}$.

Los individuos de GA_{pob} son elegidos por el operador de selección escogido (línea 7) mientras que los pertenecientes a CE_{pob} son los ejemplos con mejor fitness en Pob_t (línea 8). Una vez formadas ambas subpoblaciones, se trabaja en cada una de manera diferente:

- ◇ En GA_{pob} se aplican los operadores de cruce (línea 9) y mutación (línea 10) para la creación de los nuevos individuos.
- ◇ En CE_{pob} se utiliza un método CE para la generación de nuevos individuos a partir de la media \bar{x} y desviación σ (línea 11). Por otro lado, se escogen un número $Ejemplos_{actualizar}$ de los mejores individuos de esta subpoblación para actualizar los valores \bar{x} y σ de cara a las siguientes generaciones (línea 12).

Con los individuos resultantes de ambas subpoblaciones, nombrados en el Algoritmo 3 como $Hijos_{GA}$ e $Hijos_{CE}$ para el caso de GA_{pob} y CE_{pob} respectivamente, se crea una población completamente nueva Pob_{t+1} que sustituirá completamente a la actual Pob_t (línea 13). Por último, y tras evaluar la población Pob_{t+1} , se aplica elitismo a ésta, añadiendo el mejor individuo encontrado hasta el momento si este no se encuentra en la población (línea 15).

Como se puede comprobar, la mayoría de los parámetros que hay que tener en cuenta para el correcto funcionamiento del algoritmo son dados por el usuario. Tras realizar esta implementación, las preguntas que surgen son las siguientes: ¿Qué parámetros pueden cambiar para mejorar el rendimiento? ¿Cuál es exactamente el número de individuos necesarios para, por ejemplo, actualizar el vector de medias y desviaciones? ¿Qué porcentaje de individuos de la población hay que escoger para cada sub-población?

Para responder estas preguntas, se lleva a cabo un estudio en su correspondiente apartado del capítulo de experimentación, en el que se intenta, ya no sólo adaptar el tamaño de la población a las diferentes dimensiones del problema, sino saber qué porcentajes del tamaño total de la población deben ser para cada sub-población, y el porcentaje de individuos dentro de la población del CE que se debe usar para actualizar sus diferentes vectores. Por tanto, el algoritmo varía, quedando como resultado lo expuesto en el Algoritmo 4, publicado en [Lopez-Garcia 15, Lopez-Garcia 16b].

3. Descripción del algoritmo

Datos: P_{cruce} , $P_{mutacion}$, GA_{tam} , CE_{tam} , $Learn_{rate}$, $Ejemplos_{actualizar}$

Resultado: *Mejor individuo encontrado*

```
1  $t \leftarrow 0$ 
2  $Pob_0 \leftarrow$  Inicializar Poblacion con  $GA_{tam} + CE_{tam}$  individuos
3  $\bar{x} \leftarrow$  Inicializar Medias
4  $\sigma \leftarrow$  Inicializar Desviaciones
5 Evaluar  $Pob_0$ 
6 mientras Condición de parada no alcanzada hacer
7    $GA_{pob} \leftarrow$  Operador de Selección( $P_t$ ,  $GA_{tam}$ )
8    $CE_{pob} \leftarrow$  Seleccionar mejores ejemplos( $P_t$ ,  $CE_{tam}$ )
9    $Hijos_{GA} \leftarrow$  Operador de cruce( $GA_{pob}$ ,  $P_{cruce}$ )
10   $Hijos_{GA} \leftarrow$  Operador de mutación( $GA_{pob}$ ,  $P_{mutacion}$ )
11   $Hijos_{CE} \leftarrow$  Generar nuevos individuos( $Pob_t$ ,  $CE_{pob}$ ,  $\bar{x}$ ,  $\sigma$ )
12   $(\bar{x}, \sigma) \leftarrow$  Actualizar( $Learn_{rate}$ ,  $Ejemplos_{actualizar}$ ,  $\bar{x}$ ,  $\sigma$ )
13   $Pob_{t+1} \leftarrow Hijos_{GA} \cup Hijos_{CE}$ 
14  Añadir el mejor individuo encontrado a  $Pob_{t+1}$  si no estuviera ya
15  Evaluar  $Pob_{t+1}$ 
16   $t \leftarrow t + 1$ 
17 fin
```

Algoritmo 3: Pseudocódigo del proceso seguido por GACE cuando se aplica a problemas de predicción de la congestión.

Los cambios realizados de la primera versión a la que se presenta para la optimización de funciones matemáticas (Algoritmo 4) son los siguientes:

- ◇ Cálculo del tamaño de la población dependiendo del tamaño del problema (línea 2 del Algoritmo 4). En este caso, el tamaño del problema, se multiplicará por una constante que determinará el tamaño de la población. En la Sección 4.2 se presenta un estudio orientado a la obtención de este valor.
- ◇ Cálculo del tamaño de la sub-población GA (GA_{tam}) a partir de un porcentaje, y no como un valor fijo, adaptándose así al tamaño de la población Pob_{tam} (línea 2).
- ◇ Por consiguiente, el tamaño de la sub-población CE, CE_{tam} queda calculado como $CE_{tam} = Pob_{tam} - GA_{tam}$ (línea 3).
- ◇ El número de ejemplos utilizados para actualizar la media y desviación, $Ejemplos_{actualizar}$, se calcula ahora a partir de un porcentaje p_{up} dado por el usuario (línea 4).

Datos: P_{cruce} , $P_{mutacion}$, p_{ga} , $Learn_{rate}$, p_{up}

Resultado: *Mejor individuo encontrado*

```

1 Cálculo de  $Pob_{tam}$  a partir del tamaño del problema  $GA_{tam} \leftarrow ||Pob_{tam} \cdot p_{ga}||$ 
2  $CE_{tam} \leftarrow Pob_{tam} - GA_{tam}$ 
3  $Ejemplos_{actualizar} \leftarrow ||CE_{tam} \cdot p_{up}||$ 
4  $t \leftarrow 0$ 
5  $Pob_0 \leftarrow$  Inicializar Poblacion con  $GA_{tam} + CE_{tam}$  individuos
6  $\bar{x} \leftarrow$  Inicializar Medias
7  $\sigma \leftarrow$  Inicializar Desviaciones
8 Evaluar  $P_0$ 
9 mientras Condición de parada no alcanzada hacer
10    $GA_{pob} \leftarrow$  Operador de Selección( $P_t, GA_{tam}$ )
11    $CE_{pob} \leftarrow$  Seleccionar mejores ejemplos( $P_t, CE_{tam}$ )
12    $Hijos_{GA} \leftarrow$  Operador de cruce( $GA_{pob}, P_{cruce}$ )
13    $Hijos_{GA} \leftarrow$  Operador de mutación( $GA_{pob}, P_{mutacion}$ )
14    $Hijos_{CE} \leftarrow$  Generar nuevos individuos( $CE_{pob}, \bar{x}, \sigma$ )
15    $(\bar{x}, \sigma) \leftarrow$  Actualizar( $Learn_{rate}, Ejemplos_{actualizar}, \bar{x}, \sigma$ )
16    $Pob_{t+1} \leftarrow Hijos_{GA} \cup Hijos_{CE}$ 
17   Evaluar  $Pob_{t+1}$ 
18   Añadir el mejor individuo encontrado a  $Pob_{t+1}$  si no estuviera ya
19    $t \leftarrow t + 1$ 
20 fin

```

Algoritmo 4: Pseudocódigo del proceso seguido por GACE en la optimización de funciones matemáticas.

Los resultados de esta experimentación, así cómo los valores que finalmente se han tomado y con los que se busca ahorrar el cálculo de los mismos en cada problema en el que se aplica el algoritmo, se encuentran en la Sección 4.2, contenida en el siguiente capítulo de esta memoria.

*Las cosas no se dicen, se hacen, porque al
hacerlas se dicen solas.*

Woody Allen

4

Experimentación y resultados

Este capítulo recoge la experimentación realizada con el algoritmo propuesto en esta tesis. La Sección 4.1 cuenta con la explicación de los datasets utilizados, el cálculo de la congestión, así como la aplicación del método a las diferentes partes de los FRBSs utilizados. Por otro lado, la Sección 4.2 contiene todo lo relacionado con la aplicación de GACE a diferentes funciones matemáticas, el estudio de diferentes parámetros del algoritmo, y los resultados obtenidos por el mismo.

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

A lo largo de esta sección, se encuentra la experimentación llevada a cabo con la técnica propuesta así como información sobre los datasets utilizados, operadores y configuraciones de la misma. En la Sección 4.1.1 se encuentra la información sobre los datos utilizados, tal como su obtención, qué información aportan o cómo se van a utilizar en esta experimentación. La codificación de las soluciones se detalla en la Sección 4.1.2, mientras que los operadores utilizados tanto en la parte GA como en la parte CE se encuentran en la Sección 4.1.3. Por último, los resultados de la experimentación se muestran en la Sección 4.1.4

4. Experimentación y resultados

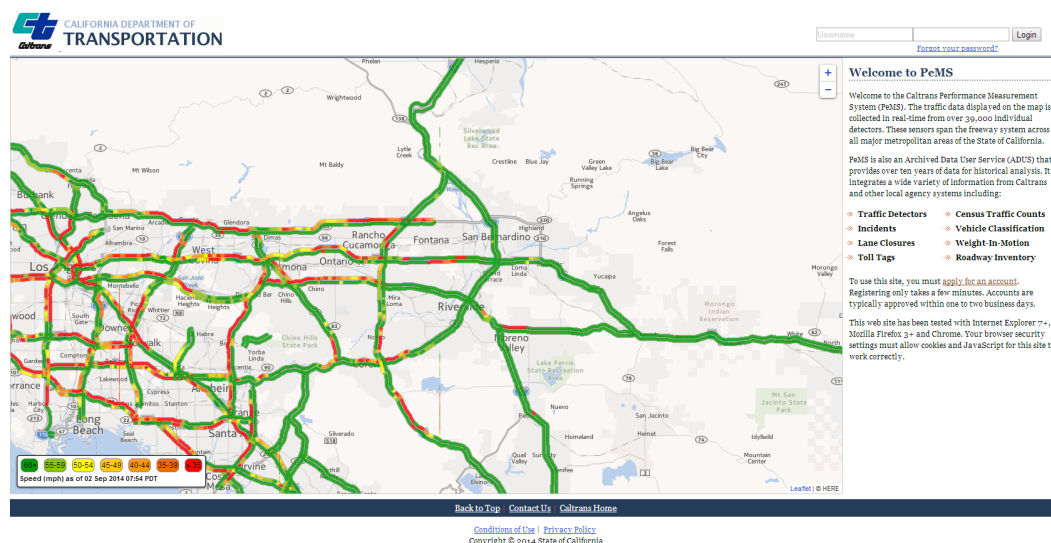


Figura 4.1: Página de Caltrans Performance Measurement System (PeMS)

4.1.1 Datos reales de tráfico

Los datos usados en este trabajo son proporcionados por el Caltrans Performance Measurement System (PeMS¹). En la Figura 4.1 puede verse una captura de pantalla de la página de la plataforma. PeMS es una base de datos en tiempo real del Departamento de Transporte de California que ofrece alrededor de 10 años de datos de tráfico para su análisis. Para este trabajo, se ha usado una sección de la carretera I5 en Sacramento, California, de 5.60 millas (aprox. 9 kilómetros).

Concretamente, el tramo elegido cuenta con 13 puntos situados en la carretera principal y 8 sensores en las entradas y salidas de la vía, de los cuáles también se han recogido datos. Las medidas de tráfico son recogidas por dichos sensores con una frecuencia de 5 minutos. La información que recogen los sensores de la carretera principal consiste en la fluidez (número de vehículos por unidad de tiempo), la ocupación (el porcentaje de tiempo que el sensor está activo) y la velocidad (en millas por hora). La información que se obtiene de los sensores situados en las entradas y salidas sólo concierne a la fluidez. Los datos se han obtenido en el intervalo de tiempo desde las 0:00 horas del 1 de Septiembre de 2013, hasta las 23:55 del 30 de Septiembre del mismo año. La Figura 4.2 muestra un esquema de la carretera así como la localización de los sensores en la misma.

Dado que sólo disponemos de información puntual, se utilizan los intervalos propuestos en [Skycomp 09] y mostrados en la Tabla 4.1 para definir el nivel de congestión en un momento dado, y poder usar estos datos para crear modelos que lo predigan. Estos intervalos están expresados en

¹ www.pems.dot.ca.gov

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

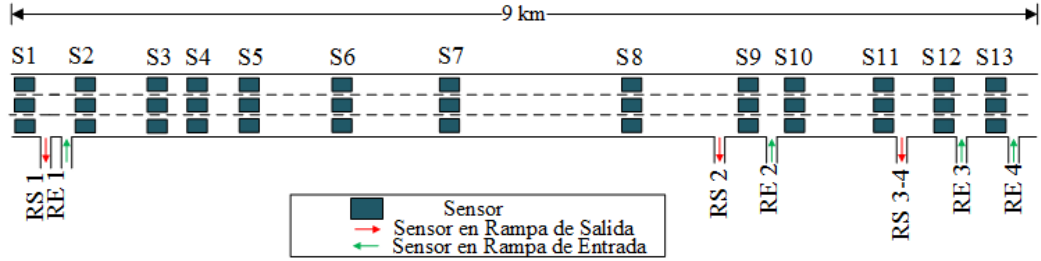


Figura 4.2: Segmento de la carretera 15. Los sensores se denotan por S , las rampas de salida por RS y las de entrada por RE .

Nivel de congestión	Densidad de Tráfico (ve/km/ln)	Velocidad del vehículo (km/h)
Ligera	[29–37]	[48–80]
Moderada	[37–50]	[24–64]
Severa	> 50	< 40
Nula	Resto de casos	

Tabla 4.1: Valores de congestión y su cálculo.

unidades del sistema internacional. Las columnas de la Tabla 4.1 muestran el nivel de congestión, la densidad del tráfico (velocidad por kilómetro y línea) y velocidad del vehículo (en kilómetros por hora). Por otro lado, destacar que, aunque la densidad de vehículos no es proporcionada por los datos de PeMS, se calcula usando los valores de velocidad y fluidez: $densidad = fluidez/velocidad$. La densidad sólo es utilizada para calcular la congestión y no se incluye como dato en los datasets.

En total, en cada instancia de un dataset completo se almacenan cada 5 minutos las siguientes variables:

- ◇ $\{F_i, O_i, V_i\}$, para todo $i = 1 \dots 13$, representando la fluidez, la ocupación y la velocidad respectivamente en el correspondiente sensor de la vía principal.
- ◇ $\{In_j \text{ e } Out_j\}$, para todo $j = 1 \dots 4$, representando la fluidez de la correspondiente entrada o salida de la vía.
- ◇ *Congestion*, como variable de clase a determinar, que puede tomar los valores $Congestion = \{Nula, Ligera, Moderada, Severa\}$ calculados según la Tabla 4.1.

Por consiguiente, contamos con un total de 48 variables: tres variables por trece sensores en la carretera principal (39), dos variables por cuatro sensores en las rampas de entrada y salida (8), y por último, la variable de clase.

4. Experimentación y resultados

4.1.1.1 Conjunto de datos

Con los datos de los que se ha hablado en la sección anterior, se han creado cuatro tipos de conjuntos de datos, o datasets:

1. *Dataset Punto (DP)*: este conjunto de datos contiene las medidas de todos los sensores. La congestión se calcula en un único punto de la carretera. En este caso, se ha escogido como dicho punto el nombrado como *S7* que puede verse en la Figura 4.2 por ser el punto medio del sector de carretera tratado.
2. *Dataset Punto Simplificado (DPS)*: este conjunto de datos es una versión simplificada de *DP*. Sólo contiene los datos del primer (*S1*), séptimo (*S7*) y último (*S13*) sensor y una combinación de los valores de fluidez de las rampas de entrada y salida. Este valor es la media de la fluidez antes y después del punto de interés. El valor de la congestión se calcula de la misma forma que en *DP*. La Figura 4.3 muestra la forma en la que se han obtenido estos datos.
3. *Dataset Sección (DS)*: este dataset contiene las medidas de todos los sensores y la congestión se calcula como el máximo nivel de congestión que se da en cualquiera de los sensores de la carretera.
4. *Dataset Sección Simplificado (DSS)*: versión simplificada de *DS*. Sólo contiene los datos del primer (*S1*), séptimo (*S7*) y último (*S13*) sensor y la media del valor fluidez de las rampas de entrada y salida. La congestión se calcula de la misma forma que en *DS*.

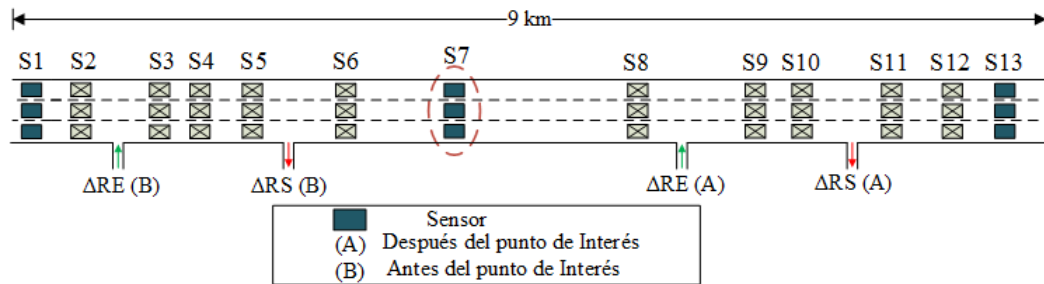


Figura 4.3: Dataset Punto Simplificado. Sólo tiene en cuenta los sensores *S1*, *S7* y *S13*, y una combinación de las rampas de entrada y salida antes y después del punto de interés *S7*.

La Tabla 4.2 contiene una clasificación de los conjuntos de datos utilizados dependiendo de los sensores de los que se han obtenido los datos y el cálculo de la congestión en cada uno de ellos.

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

Congestión Sensores	Máximo Sección	Punto
Todos	DS	DP
$S1, S3, S7$	DSS	DPS

Tabla 4.2: Clasificación de los datasets según el número de sensores utilizados y el cálculo de la congestión

El objetivo de *DP* y *DPS* es predecir la congestión en un único punto $S7$, mientras que *DS* y *DSS* pretenden predecir el máximo nivel de congestión que puede darse en la sección de carretera completa. Por otra parte, *DS* y *DP* utilizan toda la información disponible (las 47 variables más la clase anteriormente descritas), mientras que los conjuntos simplificados utilizan una versión con un menor número de atributos. En el caso de *DPS*, estos atributos son la fluidez, ocupación y velocidad medidos en los sensores antes descritos (9 variables) y cuatro valores de fluidez en las entradas y salidas antes y después del punto de interés, más la variable de clase. Por otra parte, para *DSS*, se cuenta con un total de 11 atributos sin tener en cuenta la variable clase: la fluidez, ocupación y velocidad de los sensores $S1$, $S7$ y $S13$ (9 variables), y la media ponderada de los valores de fluidez en el conjunto de rampas de entrada y salida del sector completo (2 variables).

Además, por cada tipo de dataset, para realizar la predicción de la congestión, la salida deseada será un valor de congestión a ocurrir en un horizonte de tiempo de $\{5, 10, 30\}$ minutos. Por lo tanto, se utilizarán un total de 12 conjuntos de datos que serán denotados con su acrónimo y el horizonte de tiempo. Por ejemplo, DP_5 será el correspondiente al Dataset Punto con un horizonte de 5 minutos. Otro ejemplo podría ser DSS_{15} , que correspondería al Dataset Sector Simplificado con un horizonte de predicción de 15 minutos.

Los datasets utilizados en este trabajo contienen un número muy alto de ejemplos libres de congestión con respecto a ejemplos con otros tipos de congestión. Por esta razón, se puede decir que los conjuntos de datos utilizados están altamente no balanceados. Para dar validez a dicha afirmación, se utiliza el llamado ‘*Imbalance Ratio*’ (IR) [López 13]. La Ecuación 4.1 contiene el cálculo de dicho ratio.

$$IR = \frac{MAC}{MIC} \quad (4.1)$$

donde MAC es el número de instancias de la clase mayoritaria, es decir, aquella con un mayor número de instancias, y MIC el número de instancias de la clase minoritaria, o la clase con menor número de instancias en el conjunto de datos. La Tabla 4.3 muestra el número de instancias de cada

4. Experimentación y resultados

Datasets	Nº Var	Nula	Ligera	Moderada	Severa	IR
DP_5	47	8277 → 850	61 → 61	172 → 172	129 → 129	135.6 → 13.9
DP_{15}	47	8275 → 866	61 → 61	172 → 172	129 → 129	135.6 → 14.1
DP_{30}	47	8272 → 847	61 → 61	172 → 172	129 → 129	135.6 → 13.8
DPS_5	13	8277 → 113	61 → 55	172 → 120	129 → 97	135.6 → 2.1
DPS_{15}	13	8275 → 139	61 → 45	172 → 102	129 → 103	135.6 → 3.0
DPS_{30}	13	8272 → 88	61 → 58	172 → 108	129 → 96	135.6 → 1.8

Tabla 4.3: Datasets Punto y Punto Simplificado. Los valores antes de la fecha indican su valor original, mientras que los que aparecen después se obtienen tras la reducción. Cada columna indica el tipo de congestión.

Datasets	Nº Var	Nula	Ligera	Moderada	Severa	IR
DS_5	47	4157 → 293	2776 → 473	1402 → 453	304 → 304	13.6 → 1.6
DS_{15}	47	4155 → 290	2776 → 549	1402 → 444	304 → 304	13.6 → 1.8
DS_{30}	47	4152 → 293	2776 → 464	1402 → 442	304 → 304	13.6 → 1.5
DSS_5	11	4157 → 46	2776 → 58	1402 → 65	304 → 108	13.6 → 2.3
DSS_{15}	11	4155 → 56	2776 → 57	1402 → 85	304 → 110	13.6 → 1.9
DSS_{30}	11	4152 → 54	2776 → 43	1402 → 76	304 → 114	13.6 → 2.6

Tabla 4.4: Datasets Sector y Sector Simplificado. Los valores antes de la fecha indican su valor original, mientras que los que aparecen después se obtienen tras la reducción. Cada columna indica el tipo de congestión.

tipo de congestión incluidas en cada uno de los datasets, el número de variables y el IR de cada conjunto de datos.

Con el objetivo de balancear los conjuntos de datos, se ha llevado a cabo un proceso de simplificación basado en el método de los K-Vecinos [Keller 85]. La reducción se basa en dos parámetros: k y u , donde k indica el número de vecinos escogido y u el umbral que no se puede superar por la distancia entre el nodo actual y cualquiera de los vecinos escogidos. Si este umbral es superado, se elimina el n -ésimo vecino del dataset. El método es iterativo, y se detiene cuando no hay nodos que puedan ser eliminados. Es decir, cuando ninguna de las combinaciones entre el nodo y los vecinos supere el umbral establecido. Los resultados obtenidos aplicando esta técnica a los datasets se muestran en las Tablas 4.3 y 4.4. Cada columna de la tabla muestra la cantidad de instancias de cada tipo de congestión antes y después de la aplicación de la técnica. Estas tablas también contienen el valor IR de cada conjunto para que pueda ser comparado con los datasets completos. Como se puede observar en dicha tabla, la clase mayoritaria siempre es Nula y la minoritaria Severa o Ligera antes de la simplificación. Tras ella, varía entre los distintos valores que puede tomar la congestión,

llegando incluso a ser la clase mayoritaria Severa en los datasets DSS . En el caso del valor de IR, es mucho mayor en los datasets de Punto que en los de Sección. Posteriormente a la reducción de instancias, el IR más alto corresponde a los datasets DP , mientras que, para el resto, ronda los mismos valores (entre 1.6 y 3.0).

Los datasets reducidos se usan como conjuntos de entrenamiento para el algoritmo, con la idea de evitar sobreentrenar las clases mayoritarias, mientras que los datasets completos se usan para testear los resultados.

4.1.2 Codificación de las soluciones

En esta sección, se introduce la estructura del cromosoma para poder explicar varios aspectos del algoritmo y calcular sus diferentes valores. El cromosoma codifica las tres partes que se utilizarán en cada uno de los FRBS que componen un PHFRBS:

1. *Jerarquía*: se define como el subconjunto de variables seleccionadas para ser procesadas por el PHFRBS, así como el orden en el que estas deben ser incluidas en el sistema.
2. *Funciones de pertenencia* (MF): contiene la localización de las MFs utilizadas para codificar cada una de las variables de cada sistema FRBS en la jerarquía.
3. Base de reglas (*Rule Base*, RB): codifica las posiciones de los singletons usados como consecuentes de la base de reglas de un sistema FRBS en la jerarquía.

A lo largo de esta tesis, llamaremos a estas partes $C_{jerarquia}$, $C_{etiquetas}$ y C_{reglas} respectivamente. Cada una se describe con detalle a continuación.

$C_{jerarquia}$ es una permutación en la cual un valor i en la posición j denota que la i -ésima variable se inserta en el PHFRBS en la j -ésima posición. Además, un carácter de terminación, denotado como 0, determina a partir de qué punto del vector no se utilizan más variables. Por lo tanto, el tamaño de la permutación es de $N_{variables} + 1$. El número de módulos o sistemas ($N_{modulos}$) está relacionado con $N_{variables}$: el valor máximo que puede tomar $N_{modulos}$ es $N_{variables} - 1$. Las Figuras 2.6 y 2.7, mostradas en la Sección 2.1.5, ilustran esta afirmación.

$C_{etiquetas}$ está compuesta por dos matrices llamadas MF_1 y MF_2 , cuyos valores reales se encuentran en el intervalo $[-1, 1]$, y que contienen la localización de las funciones de pertenencia de la primera y segunda variable de cada uno de los sistemas individuales FRBS respectivamente. De manera formal, $C_{etiquetas}$ está formado por dos matrices de $N_{modulos} \cdot N_{variables}$ elementos. Por lo tanto, contamos con:

4. Experimentación y resultados

$$C_{etiquetas} = MF_i(j, k) \forall i \in \{1, 2\}, \forall j \in \{1 \dots N_{modulos}\}, \forall k \in \{1 \dots N_{etiquetas}\} \quad (4.2)$$

donde cada $MF_i(j, k)$ denota la k -ésima función de pertenencia MF usada para codificar la i -ésima entrada del j -ésimo sistema FRBS en la jerarquía. Además, se usa una técnica denominada ‘*lateral tuning*’ presentada en [Alcalá 07] para la codificación de las MF. Originalmente, los valores contenidos en las MF están normalizados para ajustarse al intervalo $[-1, 1]$. Primero, se calculan las divisiones necesarias basadas en el número de etiquetas que usa el problema. Posteriormente, el método calcula la posición de las MFs en estas divisiones y convierte su valor en un intervalo $[min, max]$.

La Figura 4.4 muestra cómo trabaja esta codificación para una estructura de cuatro etiquetas. En gris aparecen las etiquetas si estuvieran uniformemente distribuidas. Podemos comprobar como, con $X_3 = 0.0$, el punto coincide con la etiqueta uniforme, mientras que, con valores negativos, como en $X_2 = -0.2$, el valor se encuentra ligeramente a la izquierda de la etiqueta correspondiente.

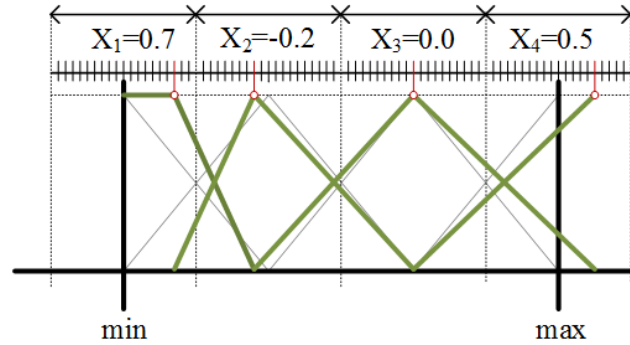


Figura 4.4: Ejemplo de codificación ‘lateral tuning’ para cuatro MFs. Cada X_i puede tomar un valor en el intervalo $[-1, 1]$.

Finalmente, se presenta C_{reglas} como una matriz de valores reales en el intervalo $[0, 1]$. Por cada sistema individual FRBS, su base de reglas RB tiene un tamaño de $N_{etiquetas}^2$. Hay que tener en cuenta que contamos con bases de reglas completas. Es decir, para cada entrada, y cada etiqueta de dicha entrada, contamos con una salida. Así pues, con un sistema i de dos entradas y tres etiquetas por entrada, las reglas, con j definida en $j \in \{1 \dots 9\}$, serían:

Si Entrada₁ = ET1₁ y Entrada₂ = ET2₁ entonces Regla_{(i)(1)}

Si Entrada₁ = ET1₁ y Entrada₂ = ET2₂ entonces Regla_{(i)(2)}

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

...

Si Entrada₁ = ET1₃ y Entrada₂ = ET2₂ entonces Regla_{(i)(j-1)}

Si Entrada₁ = ET1₃ y Entrada₂ = ET2₃ entonces Regla_{(i)(j)}

De una manera más formal, las reglas estarían definidas como sigue:

$$C_{reglas} = Reglas_i(j) \forall i \in \{1 \dots N_{modulos}\}, j \in \{1 \dots N_{etiquetas}^2\} \quad (4.3)$$

donde cada valor denota el consecuente de la regla j -ésima de la base de reglas del i -ésimo sistema FRBS en la jerarquía.

Como ejemplo, la Figura 4.5 presenta una codificación de un PHFRBS con seis variables. Como se muestra en dicho ejemplo, $C_{jerarquia}$ determina el orden en el que las variables entrarán a la jerarquía, así como las variables que serán excluidas (mediante el uso del carácter de terminación). Por otro lado, las funciones de pertenencia MF y la base de reglas RB del i -ésimo FRBS en la jerarquía están denotadas como $MF_1(i, m)$, $MF_2(i, m)$ y $Reglas_i(r)$, donde $m \in \{1, \dots, N_{etiquetas}\}$ y $r \in \{1 \dots N_{etiquetas}^2\}$.

Es importante recalcar que tanto $C_{etiquetas}$ y C_{reglas} están limitadas respectivamente a los intervalos $[-1, 1]$ y $[0, 1]$. Por lo tanto, todos los operadores implementados deben tomar los valores dentro de los correspondientes intervalos.

Hay que tener en cuenta que para la optimización de un PHFRBS se han utilizado dos codificaciones diferentes en el mismo individuo: la permutación de $C_{jerarquia}$ y los valores reales tanto de $C_{etiquetas}$ como de C_{reglas} . Por esta razón, se han usado operadores específicos que trabajan con estas codificaciones.

4.1.3 Operadores

En esta sección, se describirán los operadores utilizados en la parte genética del algoritmo propuesto (Sección 4.1.3.1), y los operadores utilizados en la parte de la CE (Sección 4.1.3.2).

4.1.3.1 Operadores utilizados en el Algoritmo Genético

En este apartado se expone la explicación de los diferentes operadores usados en la parte del GA. Es decir, la definición de los operadores de selección, cruce y mutación utilizados en esta experimentación se encuentra en esta sección.

4. Experimentación y resultados

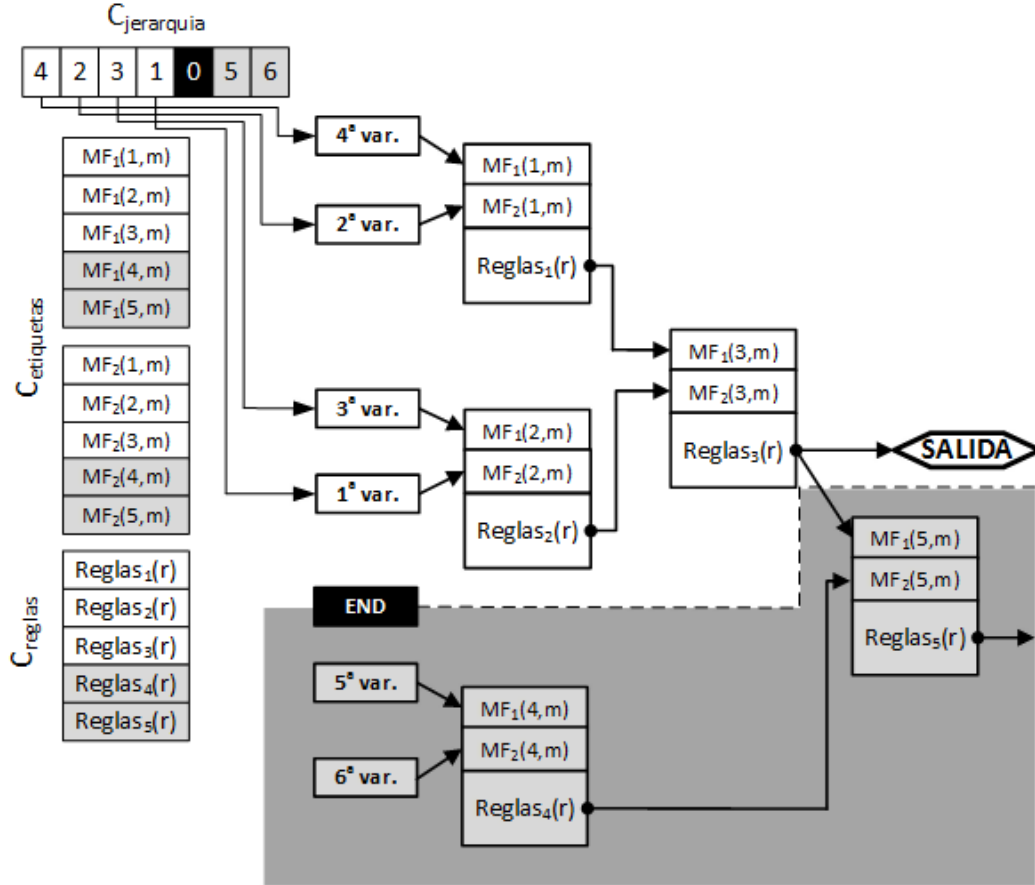


Figura 4.5: Ejemplo de codificación de un PHFRBS con seis variables de entrada.

Se usa como operador de selección el Torneo Binario [Goldberg 91]. El operador escoge dos individuos aleatoriamente dentro de la población. El ganador del torneo es el individuo con mejor fitness. El proceso se repite, en el caso que nos ocupa, hasta haber escogido GA_{tam} individuos.

Aunque las etiquetas y las reglas de nuestros individuos son matrices que contienen valores reales, la jerarquía es una permutación, como se vio en la Sección 4.1.2. Por lo tanto, son necesarios dos operadores de cruce y otros dos de mutación.

Para la permutación, se escoge una variante del Order Crossover [Deep 10]. Este operador de cruce escoge dos puntos de la permutación, dejando estático las variables que se encuentran entre los puntos y variando el resto siguiendo el orden en el que aparecen en uno de los padres. En esta variante sólo se selecciona un punto para realizar la operación. El operador escoge un punto c de manera aleatoria y conserva la permutación de los padres desde la primera posición hasta el punto elegido.

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

El resto de la jerarquía del hijo se completa de acuerdo a los valores del padre cuyo primer segmento no se ha heredado. La decisión de usar un único punto viene dada por querer mantener las primeras variables a lo largo de la herencia en su posición dado que son las que más probabilidades tienen de entrar en el PHFRBS (siempre que estén delante del carácter de terminación). La Figura 4.6 muestra un ejemplo. Considerando dos padres (P_1 y P_2), los hijos estarían formados en primera instancia por $\{2, 3, 4\}$ y $\{4, 5, 2\}$ (cajas claras). Después, empezando por el punto de corte c , los valores son escogidos en el orden dado por el otro padre, obviando los valores que ya se encuentran en los hijos, marcados con una x. Por tanto, la secuencia para O_1 es $\{1, 6, 5\}$ y para O_2 es $\{6, 1, 3\}$. Esta secuencia se coloca posteriormente al punto de corte en el hijo correspondiente.

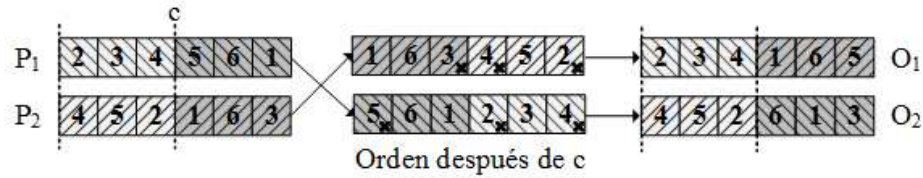


Figura 4.6: Variante del cruce de orden usada en el trabajo. Primero, se elige el punto de corte. Posteriormente, se selecciona el orden y finalmente se crean los nuevos individuos.

Para la parte real de los individuos (etiquetas y reglas), se ha escogido el método de cruce BLX- α [Herrera 98, Eshelman 93]. Dados dos padres $A = (a_1 \dots a_i)$ y $B = (b_1 \dots b_i)$ por cada i , BLX- α crea dos hijos generando valores aleatorios en el intervalo presentado en la Ecuación 4.4 con $\alpha \in [0,1]$.

$$[\min(a_i, b_i) - \alpha \cdot |a_i - b_i|, \max(a_i, b_i) + \alpha \cdot |a_i - b_i|] \quad (4.4)$$

Para los operadores de mutación, también se realiza una distinción entre jerarquía, etiquetas y reglas. Para la jerarquía, se aplica un operador de intercambio [Larranaga 99] en el que se escogen dos posiciones dentro de la permutación y estas son cambiadas la una por la otra. Para las etiquetas y las reglas, se aplica una mutación BGA [Mühlenbein 93]. Dado $A = (a_1 \dots a_m)$, el operador devuelve a_i , calculado tal y como se presenta en la Ecuación 4.5

$$a'_i = a_i \pm \beta \cdot \sum_{k=0}^{15} (\alpha_k 2^{-k}) \quad (4.5)$$

donde β define el rango de mutación. El signo (+ or -) se escoge con probabilidad 0.5, $\alpha_k \in \{0, 0.33, 0.66, 1\}$ se genera aleatoriamente con la siguiente probabilidad:

$$p(\alpha_k = \{0, 0.33, 0.66, 1\}) = \frac{1}{4}. \quad (4.6)$$

4. Experimentación y resultados

Individuo	\bar{x}_0	σ_0
$C_{jerarquia}$	$0.5 \cdot N_{variables}$	$0.5 \cdot N_{variables}$
$C_{etiquetas}$	0	1
C_{reglas}	0.5	0.5

Tabla 4.5: Valores iniciales por cada una de las partes de \bar{x} y σ .

4.1.3.2 Operadores utilizados en la Entropía Cruzada

Como se ha mencionado con anterioridad, deben mantenerse dos individuos a lo largo del CE: uno que mantenga la media \bar{x} y otro que guarde la desviación típica σ . Estos individuos tienen la misma estructura que un individuo normal (Figura 4.5), pero se inician de tal manera que generar nuevos ejemplos a partir de ellos (en la primera iteración) es equivalente a generar individuos aleatorios. Para dicho propósito, la inicialización de cada una de sus partes se realiza tal y como se presenta en la Tabla 4.5. Como valor medio se utiliza el punto medio del intervalo en el que se pueden dar los valores de cada una de las partes del individuo mientras que como desviación se toma la mitad de la amplitud de dicho intervalo. Estos valores iniciales se actualizarán durante la ejecución del algoritmo.

Los individuos que procesa el CE son seleccionados de manera determinista. Los CE_{tam} mejores individuos de la población constituyen la población CE_{pob} . Una vez que se obtengan, los individuos \bar{x} y σ actualizan sus valores y se generan los nuevos ejemplos.

La actualización se realiza aplicando directamente las ecuaciones de las líneas 7 y 8 del Algoritmo 2 a las etiquetas ($C_{etiquetas}$) y reglas (C_{reglas}) de los individuos. Dichas partes en los nuevos individuos se generan siguiendo una distribución normal con parámetros \bar{x} y σ .

Para la permutación ($C_{jerarquia}$) de los individuos, tanto la actualización de los valores media y desviación típica como la generación de los nuevos ejemplos sigue un proceso diferente. Los vectores que representan la jerarquía se convierten en vectores que guardan el orden de cada una de las variables. En el vector de orden, el valor almacenado en la posición i representa la posición que ocupa el valor i en el vector de permutación. La última posición se usa para representar la posición del carácter de terminación. Entonces, \bar{x} y σ se actualizan usando estos vectores de orden. Finalmente, se aplica una transformación inversa a los ejemplos generados: de vectores de orden a vectores de jerarquía.

Este proceso queda patente en la Figura 4.7 y funciona de la siguiente manera: dados los individuos seleccionados por el CE (a), se convierten a vectores de orden en (b) escogiendo la posición en la que se encuentra cada variable. Posteriormente, se obtienen los valores media y

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

desviación de dichas posiciones y se actualiza el valor de \bar{x} y σ (c). Por último, se generan los nuevos ejemplos a partir de dicha media y desviación en (d) y se realiza el proceso inverso de transformación de vectores de orden a vectores de jerarquía (e).

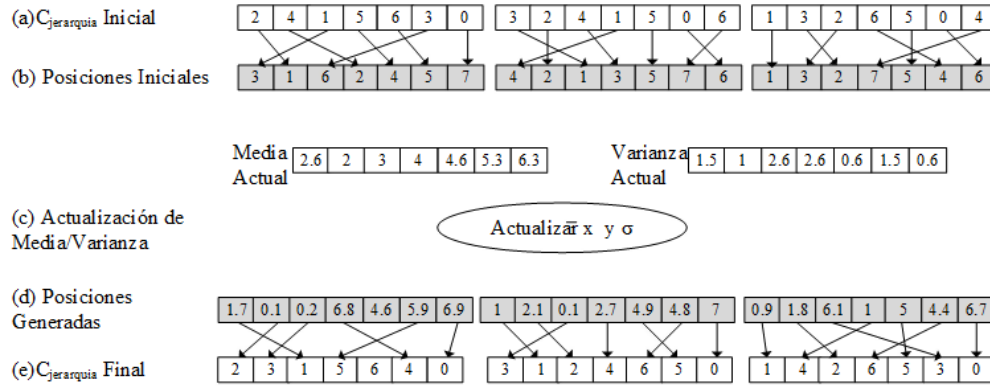


Figura 4.7: Fases de creación de la parte $C_{jerarquia}$ en un nuevo individuo.

Para explicar de manera más exhaustiva el paso de vector de jerarquía a vector de orden, se crea la Figura 4.8. Contando con un vector de jerarquía J_1 , cuyas variables se encuentran de la siguiente manera $J_1 = \{2, 4, 1, 3, 0\}$, éstas pasarían a ocupar en el vector de orden O las posiciones que ocupan en el vector J_1 , es decir, la variable 1 se encuentra en la tercera posición, la variable 2 se encuentra en la primera, y así sucesivamente. Por tanto, el vector O queda definido tal que $O = \{3, 1, 4, 2, 5\}$. Para volver a crear el vector de jerarquía a partir de un vector de orden, se realiza el paso contrario. Sea $O = \{3, 1, 4, 2, 5\}$, las variables se encontrarán en la posición que ocupan en el vector O , es decir, la variable 1 se encontrará en la posición 3 del vector de jerarquía, la variable 2 en la posición 1 y así sucesivamente. Por tanto, el vector de jerarquía J_2 será igual a $J_2 = \{2, 4, 1, 3, 0\}$. Cabe destacar que la variable de terminación establecida como 0 no tiene porqué ocupar siempre el último lugar.

Los individuos generados se combinan con los que se han creado en la parte de GA, generando la población que se procesará en la siguiente iteración del algoritmo.

4.1.4 Configuración y resultados

Se escogen combinaciones diferentes del número de individuos en GA_{pob} y CE_{pob} para realizar los experimentos y comparar sus resultados. Además, las configuraciones utilizadas en el GACE se comparan con un GA puro ($CE_{tam} = 0$) y con un CE puro ($GA_{tam} = 0$) para comprobar los beneficios de la hibridación realizada frente a la aplicación de los métodos por separado.

4. Experimentación y resultados

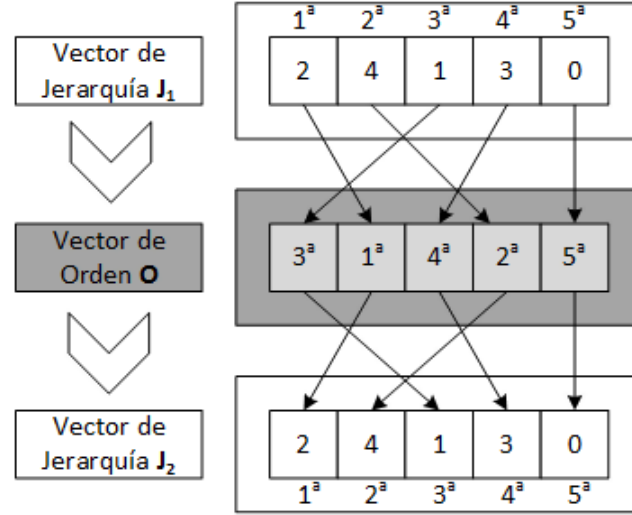


Figura 4.8: Transformación de vector de jerarquía a vector de orden y viceversa

Se llevan a cabo ocho experimentos con diferente número de individuos en la población. Cada ejecución se repite 10 veces para obtener los resultados medios. Para los experimentos, se ha establecido el número de generaciones en 500 y el tamaño total de la población en 50. Las ejecuciones se nombran por el tamaño de sus poblaciones, es decir, $GA_{tam}-CE_{tam}$. Por ejemplo, $GACE_{50-0}$ se refiere a un GA con los operadores explicados en esta memoria, y $GACE_{0-50}$ corresponde a una ejecución de una CE. El tamaño de la población del GA se encuentra en $GA_{tam} \in \{50, 45, 40, 35, 25, 15, 10, 0\}$ mientras que $CE_{tam} = 50 - GA_{tam}$.

El número de funciones de pertenencia usado en cada uno de los sistemas FRBS que componen la jerarquía está establecido a $N_{etiquetas} = 3$, por lo que el número de reglas en cada sistema individual es igual a $N_{reglas} = N_{etiquetas}^2 = 9$. El tamaño de la permutación depende del número de variables de cada conjunto de datos más uno (el carácter de terminación), y varía entre 12 y 48 (Tablas 4.3 y 4.4).

Para la parte de la hibridación correspondiente al GA, la probabilidad de cruce está establecida a 0.8 y la de mutación a 0.2. Tanto para el cruce BLX como la mutación BGA, $\alpha = \beta = 0.5$. En la parte del CE, el $Learn_{rate}$ se ha establecido a 0.7 para actualizar tanto \bar{x} como σ .

Para medir el error se aplica en primera instancia el llamado Error Medio Absoluto (*Mean Absolute Error*, MAE). Las Tablas 4.6 y 4.7 presentan el porcentaje medio de error en los conjuntos de test así como su desviación típica. Los resultados resaltados indican los dos mejores valores para cada dataset.

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

Dataset		GACE							CE
		GA	45 – 5	40 – 10	35 – 15	25 – 25	15 – 35	10 – 40	
DP_5	\overline{MAE}	0.014	0.009	0.009	0.010	0.012	0.017	0.017	0.040
	MAE_σ	0.002	0.001	0.002	0.001	0.002	0.007	0.002	0.034
DP_{15}	\overline{MAE}	0.027	0.013	0.013	0.013	0.015	0.014	0.019	0.028
	MAE_σ	0.039	0.002	0.001	0.001	0.001	0.001	0.006	0.009
DP_{30}	\overline{MAE}	0.018	0.017	0.016	0.016	0.018	0.029	0.021	0.027
	MAE_σ	0.001	0.001	0.001	0.001	0.002	0.028	0.003	0.011
DP_{S_5}	\overline{MAE}	0.012	0.011	0.008	0.009	0.010	0.033	0.058	0.220
	MAE_σ	0.006	0.003	0.001	0.002	0.002	0.043	0.097	0.133
$DP_{S_{15}}$	\overline{MAE}	0.021	0.019	0.015	0.015	0.018	0.019	0.025	0.164
	MAE_σ	0.015	0.007	0.003	0.002	0.005	0.004	0.018	0.144
$DP_{S_{30}}$	\overline{MAE}	0.015	0.015	0.015	0.018	0.014	0.017	0.020	0.158
	MAE_σ	0.002	0.002	0.001	0.011	0.001	0.004	0.006	0.129

Tabla 4.6: Media de MAE y desviación en los conjuntos de prueba de cada una de las técnicas para datasets Punto y Punto Simplificado

Se puede decir, observando dicha tabla, que en la mayoría de los casos, un GACE con $GA_{tam} \in [35, 45]$ obtiene la mejor precisión frente a otros algoritmos como $GACE_{25-25}$, GA o CE. Por lo tanto, podemos afirmar que se obtiene un rendimiento mayor por los algoritmos que usan un mayor GA_{tam} que CE_{tam} . Analizándolo a fondo, $GACE_{40-10}$ obtiene uno de los dos mejores resultados en 7 de los 12 casos mientras que $GACE_{45-5}$ lo hace en 8 de 12 casos. Por otra parte, tanto $GACE_{10-40}$ como CE no se encuentran nunca entre los mejores resultados. Comparando GA con CE, GA obtiene el mejor resultado en todos los casos mientras que CE obtiene el peor resultado en 9 de los 12 casos de estudio.

Por otro lado, MAE no tiene en cuenta cuán distantes son entre sí la predicción realizada frente a la esperada. Es decir, este error se calcula teniendo presente que la instancia predicha no es igual a la esperada, pero no cómo difiere el tipo de congestión predicho del esperado. Un ejemplo: no es lo mismo predecir una congestión Nula y que se espere una de tipo Leve a predecir una congestión Nula y que la esperada sea Severa. En este ejemplo, la segunda instancia debería obtener un mayor error que la primera.

Teniendo esto en cuenta, también se ha aplicado en segunda instancia otro tipo de error llamado sMAPE (*Symmetric Mean Absolute Percentage Error*) [Hyndman 06]. El cálculo de este error se muestra en la Ecuación 4.7, donde \bar{Y} es el valor esperado, Y el valor obtenido, y n el número de ejemplos. Usando este error, solucionamos el problema anterior.

4. Experimentación y resultados

Dataset		GACE							CE
		GA	45 – 5	40 – 10	35 – 15	25 – 25	15 – 35	10 – 40	
DS_5	\overline{MAE}	0.174	0.149	0.184	0.195	0.175	0.206	0.212	0.200
	MAE_σ	0.066	0.062	0.047	0.053	0.036	0.017	0.026	0.030
DS_{15}	\overline{MAE}	0.176	0.130	0.182	0.169	0.198	0.194	0.218	0.219
	MAE_σ	0.039	0.041	0.036	0.034	0.044	0.044	0.029	0.046
DS_{30}	\overline{MAE}	0.142	0.123	0.140	0.169	0.185	0.216	0.225	0.198
	MAE_σ	0.047	0.029	0.056	0.057	0.048	0.048	0.028	0.030
DSS_5	\overline{MAE}	0.212	0.152	0.163	0.145	0.163	0.241	0.295	0.382
	MAE_σ	0.067	0.045	0.058	0.036	0.019	0.115	0.088	0.057
DSS_{15}	\overline{MAE}	0.135	0.123	0.154	0.140	0.134	0.201	0.231	0.286
	MAE_σ	0.020	0.013	0.046	0.020	0.021	0.069	0.059	0.094
DSS_{30}	\overline{MAE}	0.132	0.134	0.135	0.139	0.133	0.132	0.208	0.320
	MAE_σ	0.026	0.012	0.015	0.011	0.009	0.021	0.065	0.111

Tabla 4.7: Media de MAE y desviación en los conjuntos de prueba de cada una de las técnicas para datasets Sector y Sector Simplificado

Dataset	GA	GACE						CE
		45 – 5	40 – 10	35 – 15	25 – 25	15 – 35	10 – 40	
DP_5	0.023	0.022	0.020	0.020	0.023	0.045	0.028	0.039
DP_{15}	0.017	0.011	0.011	0.013	0.015	0.023	0.023	0.067
DP_{30}	0.044	0.017	0.016	0.017	0.019	0.018	0.026	0.042
DPS_5	0.020	0.019	0.018	0.025	0.018	0.023	0.027	0.303
DPS_{15}	0.017	0.016	0.011	0.012	0.015	0.060	0.109	0.434
DPS_{30}	0.031	0.027	0.021	0.021	0.026	0.028	0.040	0.298

Tabla 4.8: Media del error sMAPE en los datasets DP y DPS por cada una de las técnicas.

$$sMAPE = \frac{1}{n} \sum_{i=1}^n \frac{|\overline{Y}_i - Y_i|}{(|\overline{Y}_i| + |Y_i|)/2} \quad (4.7)$$

Los mejores resultados por cada conjunto de datos se encuentran resaltados en las Tablas 4.8y 4.9. Como se ve por dichas tablas, lo obtenido anteriormente queda también constatado en estos resultados: GACE con $GA_{size} \in [35, 45]$ obtienen mejor resultado que el resto de configuraciones. $GACE_{45-5}$ obtiene esta vez uno de los dos mejores resultados en 9 de 12 casos de estudio, mientras que $GACE_{40-10}$ lo hace para 7 de los 12 casos. Tanto $GACE_{10-40}$ como CE no obtienen nunca uno de los dos mejores resultados en ningún caso.

Con estos últimos resultados se realizó una comparativa con las mejores configuraciones de

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

Dataset	GA	GACE	GACE	GACE	GACE	GACE	GACE	CE
		45 – 5	40 – 10	35 – 15	25 – 25	15 – 35	10 – 40	
DS_5	0.199	0.199	0.202	0.204	0.198	0.197	0.340	0.484
DS_{15}	0.333	0.240	0.251	0.217	0.240	0.365	0.463	0.545
DS_{30}	0.202	0.186	0.244	0.210	0.205	0.318	0.378	0.445
DSS_5	0.237	0.201	0.234	0.296	0.327	0.375	0.396	0.355
DSS_{15}	0.301	0.256	0.322	0.344	0.311	0.374	0.375	0.361
DSS_{30}	0.306	0.221	0.328	0.299	0.346	0.343	0.387	0.387

Tabla 4.9: Media del error sMAPE en los datasets DS y DSS por cada una de las técnicas.

GACE encontradas y diversas técnicas de la literatura. Estas técnicas fueron:

- ◇ C4.5 [Quinlan 14] es un algoritmo que genera árboles de decisión a partir de un conjunto de ejemplos proporcionados. Dicho árbol de decisión se construye desde arriba hacia abajo. Es una mejora del algoritmo ID3[Quinlan 86].
- ◇ Linear Decreasing Weight - Particle Swarm Optimization (LDWPSO) [Sousa 04] es un algoritmo PSO que usa como función de velocidad la función de pesos decrecientes lineales, de donde recibe su nombre.
- ◇ Steady-State Genetic Algorithm for Extracting Fuzzy Classification Rules (SGERD) [Mansoori 08] es un algoritmo genético de tipo steady-state (no crea una nueva población al término de la generación). Las generaciones durante las que se aplica el algoritmo son finitas y están ligadas a la dimensión del problema. La selección de los individuos no es aleatoria y sólo sobreviven los mejores. La función fitness utilizada por este método se basa en el criterio de evaluación de reglas. SGERD se detiene cuando ningún hijo es incluido en la población.
- ◇ Adaboost [del Jesus 04] es un algoritmo de boosting, el cual invoca repetidamente un algoritmo de aprendizaje para generar clasificadores difusos simples y de baja calidad. Cada vez que uno de estos clasificadores se añade al conjunto, los pesos de los ejemplos en el conjunto de entrenamiento cambian y se le otorga al clasificador en cuestión un valor determinado, el cual dependerá de su precisión. En este algoritmo, cada hipótesis se trata como una regla difusa extraída de los datos.
- ◇ Grammar-based Genetic Programming Algorithm, (GP) [Sánchez 01], se usa para entrenar un clasificador difuso a partir de la media de las reglas usando algoritmos de orogramación genética.

4. Experimentación y resultados

	GACE								
Dataset	45 – 5	40 – 10	35 – 15	C4.5	LDWPSO	SGERD	AdaBoost	GP	INNER
DP_5	0.022	0.020	0.020	0.002	0.100	0.043	0.042	0.014	0.028
DP_{15}	0.011	0.011	0.013	0.004	0.100	0.043	0.042	0.017	0.027
DP_{30}	0.017	0.016	0.017	0.004	0.089	0.043	0.042	0.020	0.036
DPS_5	0.019	0.018	0.025	0.018	0.013	0.604	0.016	0.011	0.073
DPS_{15}	0.016	0.011	0.012	0.033	0.027	0.108	0.062	0.023	0.030
DPS_{30}	0.027	0.021	0.021	0.052	0.027	0.029	0.032	0.025	0.027
DS_5	0.199	0.202	0.204	0.048	0.606	0.202	0.433	0.333	0.169
DS_{15}	0.240	0.251	0.217	0.140	0.604	0.396	0.386	0.254	0.227
DS_{30}	0.186	0.244	0.210	0.176	0.602	0.395	0.392	0.409	0.205
DSS_5	0.201	0.234	0.296	0.362	0.362	0.684	0.360	0.249	0.722
DSS_{15}	0.256	0.322	0.344	0.328	0.423	0.766	0.429	0.266	0.602
DSS_{30}	0.221	0.328	0.299	0.310	0.400	0.270	0.566	0.241	0.216

Tabla 4.10: Comparativa de las configuraciones de GACE con los algoritmos del estado del arte

- ◇ INNER [Luaces 03] es un algoritmo que intenta extraer un conjunto pequeño de reglas para representar correctamente el conjunto de entrenamiento, obteniendo una precisión aceptable.

Dichas técnicas se han escogido, no sólo por su uso en la literatura actual y sus diferentes mecánicas, también por su facilidad de reproducir, en el caso de que hiciera falta, la experimentación a la vez que el fácil acceso a sus implementaciones. Los resultados de esta comparativa se muestran en la Tabla 4.10. En los casos donde se pretende predecir la congestión en un punto (datasets DP y DPS), C4.5 es mejor que GACE en 3 de los 6 casos de estudio de estos conjuntos de datos, mientras que GACE está presente como una de las dos mejores técnicas en 4 de los 6 conjuntos estudiados, siendo superado por el susodicho C4.5 y GP en el caso del dataset DP_5 . En el caso de los datasets de tipo sector (DS y DSS), GACE obtiene mejores resultados que los métodos con los que se compara en 4 de los 6 casos de estudio propuestos. Por otro lado, C4.5 resulta ser la mejor técnica en 2 de estos casos, mientras que INNER obtiene uno de los resultados resaltados en 3 de 6 casos. Por su lado, GP obtiene 4 resultados destacados en el total de los 12 casos de estudio, mientras que LDWPSO obtiene sólo uno de los dos mejores errores en los 12 casos. Tanto Adaboost como SGERD no obtienen ningún resultado destacado.

Una vez realizada la comparativa, y viendo los resultados obtenidos por la técnica propuesta, es importante conocer qué porcentaje de cada tipo de congestión ha predicho con éxito cada una de las diferentes configuraciones. Para ello, las Figuras 4.9, 4.10, 4.11, y 4.12 muestran el porcentaje de instancias clasificadas correctamente por cada tipo de congestión que se ha predicho en este trabajo. Cada barra indica los valores de media y desviación de cada ejecución. Por lo tanto, hay 8 barras por

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

cada imagen. Cada barra se denota por su GA_{tam} en el eje X . Por lo tanto, de izquierda a derecha los valores van desde el GA puro hasta el CE.

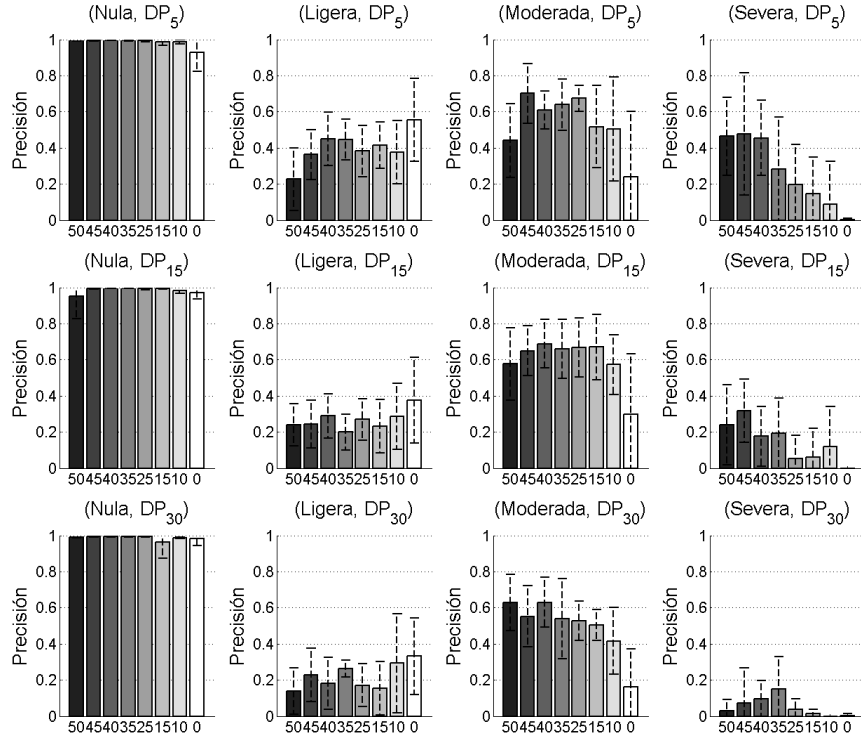


Figura 4.9: Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DP_5 (arriba), DP_{15} (centro), y DP_{30} (abajo).

Si se analizan estas figuras, podemos decir que:

- ◇ En la mayoría de los casos, las configuraciones del GACE mejoran o igualan el rendimiento obtenido por las técnicas que lo forman, GA y CE.
- ◇ De las dos partes, el GA puro es el que muestra unos valores de error más cercanos a los obtenidos por GACE en la mayoría de los conjuntos.
- ◇ Los mejores valores de predicción se obtienen cuando el margen de tiempo es el más pequeño de los presentados (5 minutos) y cuando se quiere predecir la congestión en un punto (datasets DP y DPS).
- ◇ Para los datasets DS y DSS ocurre todo lo contrario: una previsión de 30 minutos obtiene buenos resultados.

4. Experimentación y resultados

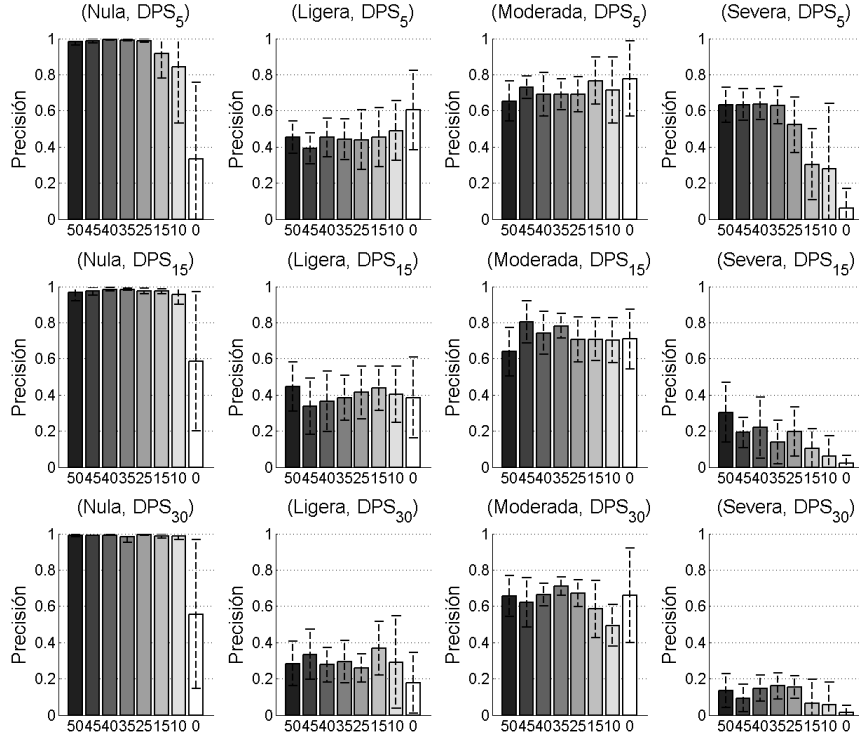


Figura 4.10: Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DPS_5 (arriba), DPS_{15} (centro), y DPS_{30} (abajo).

- ◊ Para los conjuntos de datos simplificados, se obtiene prácticamente el mismo valor para cualquier tipo de congestión, siendo los mejores resultados los obtenidos en DPS_5 .
- ◊ Si se predice congestión en un segmento (datasets DS y DSS), se obtienen buenos resultados a la hora de predecir congestión Severa en cualquier dataset.
- ◊ Predecir congestión Severa en un segmento obtiene valores significativamente mejores que predecir congestión Nula en conjuntos de datos simplificados, especialmente en DSS_5 y DSS_{15} , donde los resultados entre los diferentes tipos de congestión son más estables.

Con los datos de la Tabla 4.6, donde se utiliza MAE como métrica del error, se ha utilizado el conocido test t de Student para mostrar las diferencias estadísticas que existen entre cada una de las configuraciones y algoritmos utilizados. En dicho test, utilizamos los siguientes valores:

1. $(++)$ en el caso de que los resultados del algoritmo A_1 sean significativamente mejores a los del algoritmo A_2 .

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

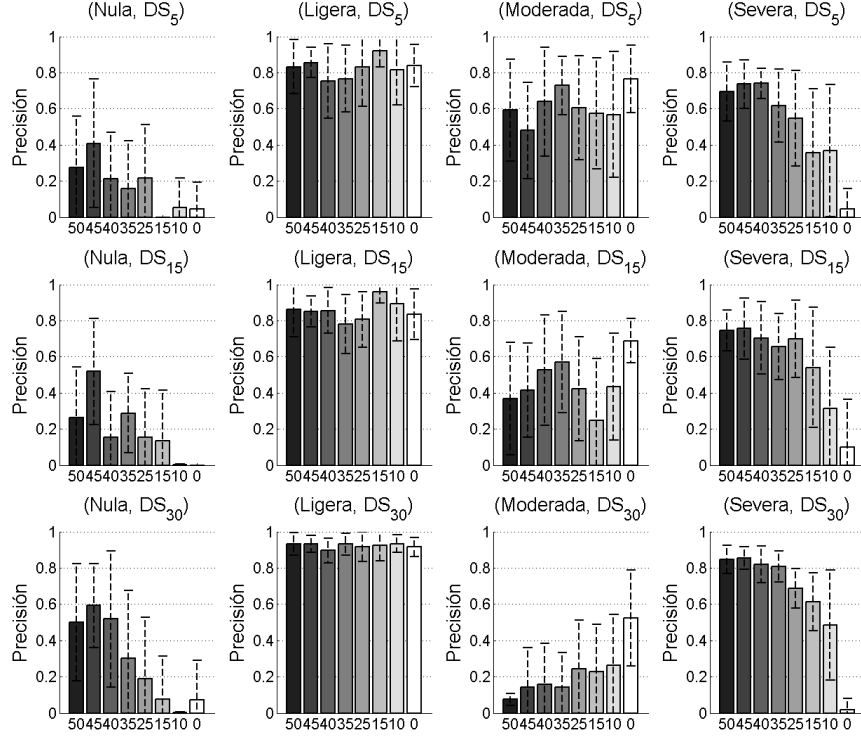


Figura 4.11: Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DS_5 (arriba), DS_{15} (centro), y DS_{30} (abajo).

2. (+) en el caso de que los resultados del algoritmo A_1 sean mejores en promedio, aunque no significativamente mejores, a los del algoritmo A_2 .
3. (−) en el caso de que los resultados del algoritmo A_1 sean peores a los obtenidos por el algoritmo A_2 .
4. (−−) en el caso de que los resultados del algoritmo A_1 sean significativamente peores a los obtenidos por el algoritmo A_2 .

El cálculo de este valor se realiza siguiendo la Ecuación 4.8:

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{(n_1-1)SD_1^2 + (n_2-1)SD_2^2}{n_1+n_2-2} \frac{n_1+n_2}{n_1n_2}}} \quad (4.8)$$

Donde \overline{X}_i , SD_i y n_i , son la media, la desviación típica y el número de ejecuciones de cada técnica. En nuestro caso, n_i está definido con un valor 10 para todas las técnicas, mientras que los valores medios y de desviación son los obtenidos en la Tabla 4.6. El intervalo de confianza escogido es de 0.95.

4. Experimentación y resultados

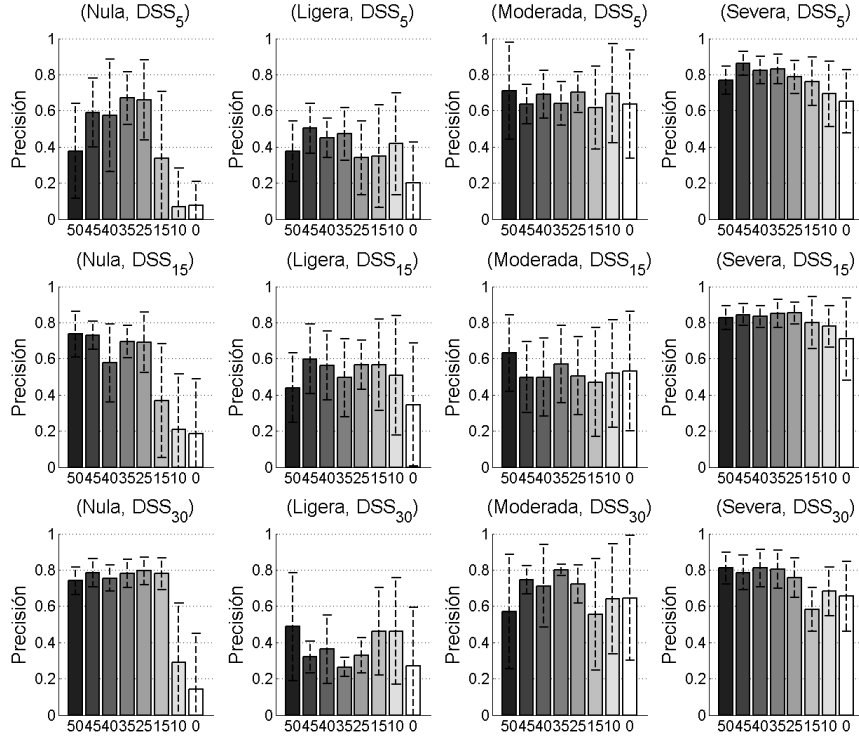


Figura 4.12: Porcentaje de instancias correctamente clasificadas por cada clase en los datasets DSS_5 (arriba), DSS_{15} (centro), y DSS_{30} (abajo).

Como resultado, se incluye en las Tablas 4.11 y 4.12 el porcentaje de veces que los resultados de un algoritmo son mejores (+) o significativamente mejores que los de otro (++), quedando demostrado lo observado en la experimentación anterior. Realizando un análisis de dichas tablas, podemos decir que:

- ◇ La configuración $GACE_{45-5}$ es mejor o significativamente mejor que el GA puro en más del 80 % de los casos.
- ◇ Por otro lado, $GACE_{45-5}$ es significativamente mejor (++) que el CE puro en todos los casos expuestos.
- ◇ Comparando configuraciones de la propuesta, utilizando $GACE_{40-10}$ se mejora el rendimiento obtenido por $GACE_{45-5}$ en un 25 % de los casos totales, mientras que la aplicación de $GACE_{45-5}$ supera a $GACE_{40-10}$ en el 50 % de los 48 casos.
- ◇ Por otro lado, la configuración $GACE_{35-15}$ supera a $GACE_{40-10}$ en el mismo porcentaje total de casos en los que $GACE_{40-10}$ es mejor a $GACE_{45-5}$, un 25 %.

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

	GA	$GACE_{45-5}$	$GACE_{40-10}$	$GACE_{35-15}$	$GACE_{25-25}$	$GACE_{15-35}$	$GACE_{10-40}$	CE
GA	X	8,33	33,33	41,67	33,3	58,33	16,67	16,67
$GACE_{45-5}$	50	X	41,67	25	33,3	33,33	16,67	0
$GACE_{40-10}$	41,67	8,33	X	50	25	58,33	25	16,67
$GACE_{35-15}$	33,3	25	25	X	41,67	41,67	33,33	16,67
$GACE_{25-25}$	41,67	25	25	33,3	X	33,3	33,33	25
$GACE_{15-35}$	16,67	8,33	8,33	16,67	16,67	X	66,67	16,67
$GACE_{10-40}$	8,33	0	0	0	0	8,33	X	25
CE	0	0	0	0	0	25	16,67	X

Tabla 4.11: Porcentaje de veces en los que los resultados de un algoritmo son mejores que los obtenidos por otro (+)

- ◇ A su vez, $GACE_{45-5}$ mejora a $GACE_{35-15}$ en un porcentaje superior que a $GACE_{40-10}$ con casi un 60 % de los casos de estudio.
- ◇ Comparando los resultados obtenidos por las técnicas puras, GA es mejor o significativamente mejor que CE en el 100 % de los casos.
- ◇ La aplicación de un GA puro es únicamente significativamente mejor (++) que las configuraciones del algoritmo $GACE_{15-35}$, $GACE_{10-40}$, y CE , y obtiene mejor resultado (+) en un porcentaje menor del 50 % cuando es comparada con el resto de configuraciones, siendo el porcentaje más alto obtenido al compararlo con la configuración $GACE_{35-15}$ con aproximadamente un 42 %.
- ◇ $GACE_{10-40}$ y $GACE_{15-35}$ obtienen un resultado de ++ comparándolos con CE en un 58 % de los casos y, a su vez, $GACE_{15-35}$ obtiene un resultado de + y ++ comparándolo con la configuración $GACE_{10-40}$ en más del 80 % de los casos totales.
- ◇ CE sólo es mejor que las configuración $GACE_{15-35}$ en un 25 % de los casos y que $GACE_{10-40}$ en un 16 %.
- ◇ Por último, CE no obtiene ningún valor ++ cuando es comparado con alguno de las configuraciones propuestas.

Para ilustrar lo obtenido por el test de Student, las Figuras 4.13, 4.14, 4.15 y 4.16 muestran los resultados generales. Los colores verde muestran cuando un algoritmo del eje Y es mejor que otro en el eje X, siendo el color verde más oscuro el correspondiente al valor (+) y el más claro a (++). Igualmente, cuando un algoritmo es peor estadísticamente que otro, se muestran colores rojo que

4. Experimentación y resultados

	GA	$GACE_{45-5}$	$GACE_{40-10}$	$GACE_{35-15}$	$GACE_{25-25}$	$GACE_{15-35}$	$GACE_{10-40}$	CE
GA	X	0	0	0	0	16,67	66,67	83,33
$GACE_{45-5}$	33,33	X	8,33	33,33	33,33	50	83,33	100
$GACE_{40-10}$	16,67	16,67	X	0	33,33	33,33	75	83,33
$GACE_{35-15}$	25	8,33	0	X	25	41,67	66,67	83,33
$GACE_{25-25}$	16,67	0	8,33	0	X	41,67	66,67	75
$GACE_{15-35}$	0	0	0	0	8,33	X	16,67	58,33
$GACE_{10-40}$	0	0	0	0	0	0	X	58,33
CE	0	0	0	0	0	0	0	X

Tabla 4.12: Porcentaje de veces en los que los resultados de un algoritmo son significativamente mejores que los obtenidos por otro (++)

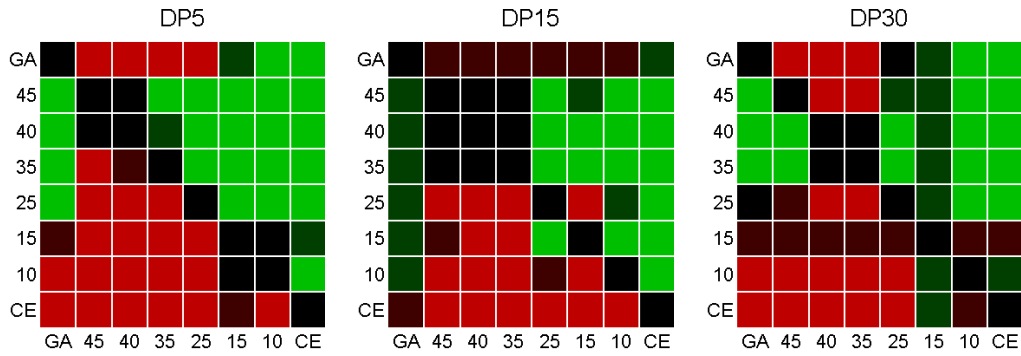


Figura 4.13: Resultados del Test de Student para los datasets DP .

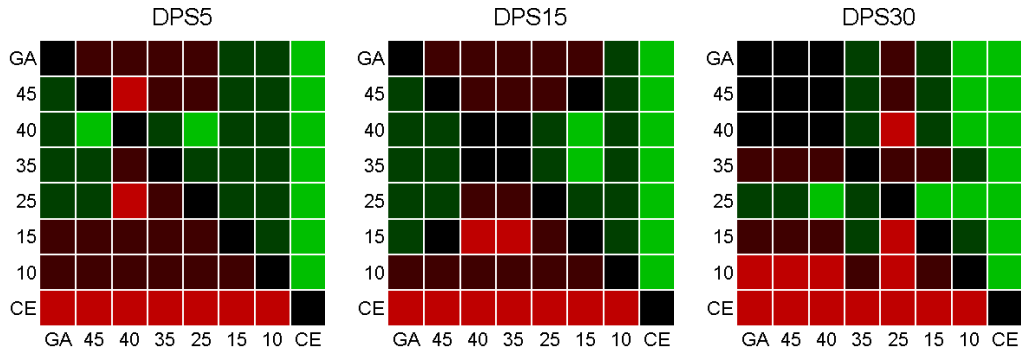


Figura 4.14: Resultados del Test de Student para los datasets DPS .

corresponden a $(-)$ el más oscuro y el más claro a $(--)$. Por último, cuando los resultados de ambos algoritmos son iguales, se muestran casillas de color negro.

En estas figuras queda patente lo obtenido en la Tabla 4.6, donde las técnicas $GACE_{45-5}$ y $GACE_{40-10}$ obtienen por lo general los mejores resultados. Se puede comprobar como en todas estas figuras, su parte baja suele ser roja, lo que nos indica que las técnicas con bajo nivel de GA son

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

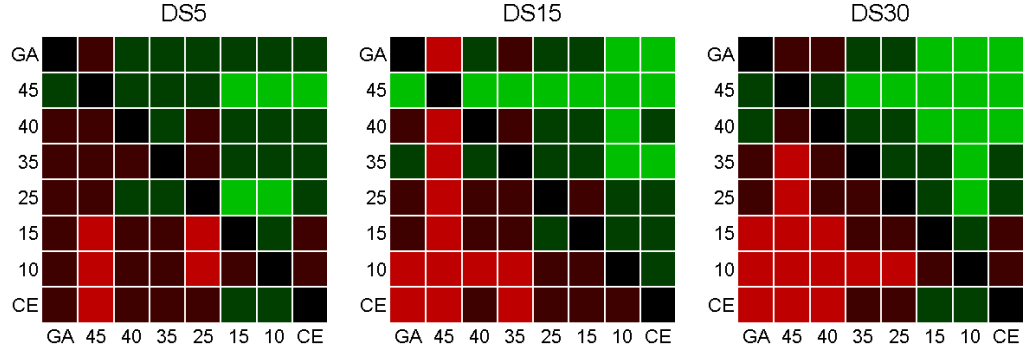


Figura 4.15: Resultados del Test de Student para los datasets DS .

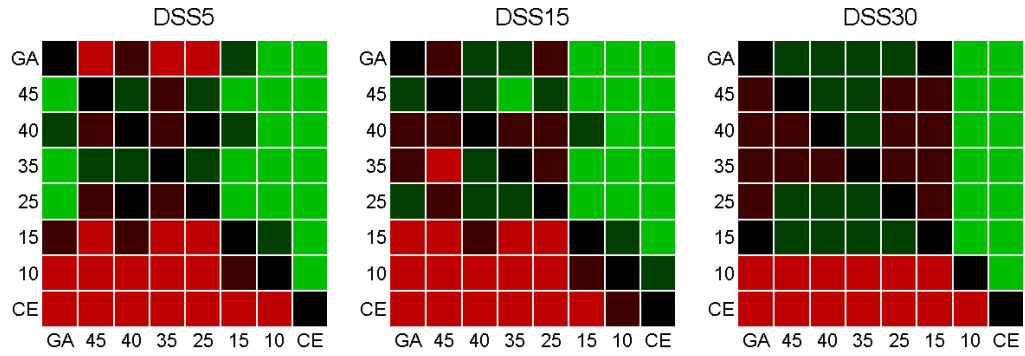


Figura 4.16: Resultados del Test de Student para los datasets DSS .

normalmente mejoradas por el resto.

Por otro lado, la Tabla 4.13 muestra la posición que ocupa cada técnica en todos los datasets, es decir, la posición media que ocupa cuando son ordenadas por mejor resultado obtenido en cada conjunto. Los resultados muestran que:

- ◇ $GACE_{45-5}$ es el mejor situado en 2 de los 4 tipos de congestión, cuando esta toma los valores Nula y Severa.
- ◇ $GACE_{40-10}$ se encuentra entre los dos mejores casos en los tipos de congestión Nula, Moderada y Severa, "fallando" únicamente en la congestión Ligera.
- ◇ $GACE_{35-15}$ acompaña a $GACE_{40-10}$ en un único caso, siendo mejor que este para predecir congestión Moderada.
- ◇ Las configuraciones $GACE_{15-35}$ y $GACE_{10-40}$ se alzan como mejor técnica a utilizar en el caso de predecir congestión Ligera.

4. Experimentación y resultados

		GACE	GACE	GACE	GACE	GACE	GACE	
	GA	45 – 5	40 – 10	35 – 15	25 – 25	15 – 35	10 – 40	CE
Nula	4.16	2.66	2.91	3.08	3.08	5.75	6.75	7.58
Ligera	4.50	4.25	4.83	5.33	5.33	3.66	3.41	4.66
Moderada	5.33	4.66	3.58	3.16	3.91	5.75	5.41	4.16
Severa	2.91	2.33	2.58	3.16	4.16	6.25	6.75	7.83
Total	4.22	3.47	3.47	3.68	4.12	5.35	5.58	6.06

Tabla 4.13: Media de posición para cada técnica cuando predice diferentes niveles de congestión. Los dos mejores valores están resaltados para cada caso.

	Nula	Ligera	Moderada	Severa
DP_5	$GACE_{40-10}$	CE	$GACE_{45-5}$	$GACE_{45-5}$
DP_{15}	$GACE_{35-15}$	CE	$GACE_{40-10}$	$GACE_{45-5}$
DP_{30}	$GACE_{25-25}$	CE	GA	$GACE_{35-15}$
DPS_5	$GACE_{40-10}$	CE	CE	$GACE_{40-10}$
DPS_{15}	$GACE_{40-10}$	GA	$GACE_{45-5}$	GA
DPS_{30}	$GACE_{25-25}$	$GACE_{15-35}$	$GACE_{35-15}$	$GACE_{35-15}$
DS_5	$GACE_{45-5}$	$GACE_{15-35}$	CE	$GACE_{40-10}$
DS_{15}	$GACE_{45-5}$	$GACE_{15-35}$	CE	$GACE_{45-5}$
DS_{30}	$GACE_{45-5}$	$GACE_{10-40}$	CE	$GACE_{45-5}$
DSS_5	$GACE_{35-15}$	$GACE_{45-5}$	GA	$GACE_{45-5}$
DSS_{15}	GA	$GACE_{45-5}$	GA	$GACE_{25-25}$
DSS_{30}	$GACE_{25-25}$	GA	$GACE_{35-15}$	GA

Tabla 4.14: Mejores configuraciones de GACE que predicen congestión para cada conjunto de datos.

4.1 Aplicación del algoritmo a problemas de predicción de tráfico

- ◇ CE queda como última técnica a elegir en 2 de los 4 casos, con congestión Nula y Severa, mientras que en los dos restantes la última posición es ocupada por las configuraciones $GACE_{35-15}$ y $GACE_{25-25}$ en el caso de congestión Ligera, y $GACE_{15-35}$ en el caso de congestión Moderada.
- ◇ En líneas generales, $GACE_{45-5}$ y $GACE_{40-10}$ están mejor situados para ser utilizados que el resto de configuraciones, mientras que GA sería la cuarta opción por encima de las configuraciones $GACE_{15-35}$, $GACE_{10-40}$ y CE.

Para obtener más detalles de la experimentación realizada en este apartado, se presenta la Tabla 4.14, donde se muestran las mejores configuraciones para cada dataset y congestión. Analizando dicha tabla, podemos llegar a las siguientes conclusiones:

- ◇ Las configuraciones del algoritmo propuesto obtiene mejores resultados que GA y CE en 32 de 48 casos, lo que representa que es superado por uno de los dos métodos puros en sólo el 33 % de los casos.
- ◇ Haciendo una división por tipo de congestión, GACE es la técnica a utilizar en congestión Nula en 11 de 12 casos y en Severa en 10 de 12, mientras que, para congestión Ligera y Moderada, GACE es la técnica elegida en 6 y 5 casos respectivamente.
- ◇ Destaca que $GACE_{15-35}$ sea el elegido en 3 casos de congestión Ligera en diferentes datasets dado los resultados obtenidos a lo largo de la experimentación, donde las técnicas con mayor población en su parte genética obtenían mejores resultados.
- ◇ CE, pese a su mal rendimiento durante toda la experimentación, es el algoritmo elegido en un total de 8 de 42 casos, centrándose en los datasets completos de Punto y Sector en los tipos de congestión Ligera y Moderada. GA, por su parte, obtiene los mismos resultados: es elegida 8 veces, sin un patrón establecido.
- ◇ La técnica que obtiene mejores resultados un mayor número de veces ha sido GACE con población $GACE_{45-5}$, con un total de 12 de 48 casos, de los cuáles 3 son en congestión Nula, 2 en Ligera, 2 en Moderada y 5 en Severa, siendo en este último tipo de congestión la más elegida.
- ◇ En el caso opuesto, $GACE_{10-40}$ es elegida únicamente una vez, en detección de congestión Ligera en un dataset de tipo Sector, lo que la convierte en la técnica menos elegida de todo el conjunto presentado.

4. Experimentación y resultados

- ◇ Finalmente, y como dato puntual, todas las técnicas han sido elegidas al menos una vez a lo largo de los 48 casos de estudio.

Como conclusiones a este apartado de la experimentación realizada, podemos determinar los siguientes hechos:

- ◇ El uso del algoritmo propuesto con población genética en el intervalo $GA_{tam} = [35, 45]$ aporta mejores resultados que el resto de configuraciones utilizadas y los algoritmos puros, ya sea utilizando error MAE o $sMAPE$.
- ◇ En la comparativa con las técnica propuestas, GACE, utilizando las configuraciones mencionadas en el punto anterior, obtiene uno de los dos errores con menor valor en todos los datasets.
- ◇ En lo que a instancias de los diferentes tipos de congestión bien clasificadas se refiere, GACE mejora o iguala el rendimiento obtenido por los algoritmos puros GA y CE.
- ◇ Estadísticamente hablando, las configuraciones $GACE_{45-5}$, $GACE_{40-10}$ y $GACE_{35-15}$ se encuentran entre las técnicas más destacadas obteniendo resultados mejores o significativamente mejores frente al resto de configuraciones y técnicas.
- ◇ Mientras que $GACE_{45-5}$ es la mejor técnica para poder predecir congestión Nula y Severa, $GACE_{15-35}$ y $GACE_{35-15}$ lo son para congestiones de tipo Ligera y Moderada respectivamente.
- ◇ Por último, y en términos generales, de los 48 casos de estudio, GACE, cualquiera que sea su configuración, obtiene resultados por encima de los métodos puros en 32 de los casos, siendo por tanto superado únicamente en un 33 % de los casos totales.

4.2 Aplicación del algoritmo a funciones de optimización

Esta sección se encarga de recoger toda la experimentación relacionada con la aplicación del algoritmo propuesto en esta tesis a funciones de optimización. Las funciones que se han utilizado quedan definidas en la Sección 4.2.1. Para poder contar con unos valores por defecto y realizar las modificaciones de las que se ha hablado en la Sección 3.6, se realiza un estudio de los parámetros en la Sección 4.2.2. Por último, la comparativa con otras técnicas de la literatura se encuentra contenida en la Sección 4.2.3.

4.2.1 Funciones utilizadas

Un total de 24 funciones de optimización extraídas de la plataforma Comparing Continuous Optimisers (COCO) se han utilizado para esta parte de la experimentación. Esta plataforma se utiliza en el workshop Black Box Optimization Benchmarking (BBOB). En dicho workshop, que tiene lugar durante las conferencias internacionales GECCO y CEC desde 2009 en adelante, se aplican algoritmos propuestos por investigadores a las funciones anteriores con diferente número de dimensiones. El objetivo de esta competición es probar estos algoritmos en funciones benchmark, de cara a encontrar el mejor valor posible para la función en un tiempo predefinido, tal y como se explicó en la Sección 2.3.3.

Estas funciones están libres de ruido y son de diferentes tipos. Para no entorpecer la lectura del documento, así como mantener la atención en el capítulo de experimentación que nos ocupa, el nombre, tipo y fórmula de estas funciones se adjuntan en el Apéndice B. Las funciones se denotarán a lo largo de este capítulo con la nomenclatura F_i , donde i toma valores desde 1 hasta 24, siendo estas de los siguientes tipos:

- ◇ De la función F_1 hasta la función F_5 , las funciones son de tipo separable.
- ◇ Se cuenta con un total de 4 funciones con condicionamiento bajo o moderado, correspondientes a los índices F_6 hasta F_9 .
- ◇ Las funciones desde la F_{10} hasta la función F_{14} serán de tipo condicionamiento elevado y unimodales.
- ◇ Las funciones multimodales con estructura global adecuada se encuentran en las funciones desde F_{15} hasta F_{19} .
- ◇ Por último, las funciones de F_{20} hasta F_{24} serán de tipo multimodal con estructura global débil.

Por tanto, contamos en total con 5 funciones separables, 4 funciones con condicionamiento bajo o moderado, 5 con condicionamiento elevado y unimodales, 5 funciones multimodales con estructura global adecuada y, por último, 5 funciones multimodales con estructura global débil.

La razón de usar estas funciones, como ya se ha expuesto, es demostrar la adaptabilidad de GACE a problemas de optimización genéricos, para así demostrar que puede utilizarse de una manera sencilla en otros marcos de trabajo, garantizando unos niveles de calidad adecuados en comparación con técnicas de optimización encontradas en el estado del arte. Por otra parte, se busca reducir el

4. Experimentación y resultados

Parámetros	Valor
POB_{tam}	$\{2, 5, 10, 20\} \cdot dim$
Porcentaje de individuos del GA	$p_{ga} = \{0.25, 0.5, 0.75\}$
Porcentaje de individuos del CE	$p_{ce} = 1 - p_{ga}$
Parámetros del GA	$p_c = 0.9, p_m = \frac{1}{dim}, \alpha = 0.5$
Parámetros de CE	$L_r = 0.7, p_{up} = 1$

Tabla 4.15: Valores de los diferentes parámetros utilizados en la experimentación

número de parámetros a introducir por el usuario, como es el tamaño de la población, o la tasa de actualización del CE, y estableciendo unos valores por defecto al resto de parámetros de la configuración, los cuales deben garantizar un grado de calidad en las soluciones.

Por último, se han modificado el operador de mutación con respecto a la experimentación anterior, cambiando el algoritmo BGA por un algoritmo Gaussiano, manteniendo como operador de cruce el algoritmo BLX- α .

4.2.2 Estudio de los parámetros del algoritmo

En esta sección se detalla el estudio de los diferentes parámetros de los que consta el algoritmo de forma que se pueda reducir su número. En primer lugar, se buscará definir unos valores por defecto para dichos parámetros. Para poder fijar el tamaño de la población de cara a futuras experimentaciones e investigaciones, este modifica su valor dependiendo del valor dado a la dimensión del problema. Por tanto, se ha definido el tamaño de población como $POB_{tam} = c \cdot dim$, donde c es una constante que puede tomar los valores $c = \{2, 5, 10, 20\}$, y dim la dimensión, que corresponde con el tamaño del individuo.

Para cada uno de los posibles valores de POB_{tam} , se tienen en cuenta diferentes porcentajes asociados al tamaño de población genética GA_{tam} . Este parámetro, definido como p_{ga} , toma el 25 %, 50 % y 75 % del tamaño total de la población POB_{tam} , es decir, $p_{ga} = \{0.25, 0.5, 0.75\}$. En la Tabla 4.15 se resumen los valores por defecto que se han tomado para los diferentes parámetros utilizados en esta experimentación.

En primer lugar, es necesario determinar el valor de la constante c para cada posible valor de la dimensión dim , dado que esto determinará el tamaño de la población en cada una de las dimensiones en las que vamos a trabajar. Siendo dim una variable que puede tomar los valores $dim = \{5, 10, 20, 40\}$, el tamaño de la población quedará definido en el intervalo $POB_{tam} \in [10, 800]$. A su vez, tener una primera aproximación al valor que tiene que tomar p_{ga} nos ofrecerá una idea general de los tamaños de las sub-poblaciones. Teniendo en cuenta los valores que p_{ga} puede

4.2 Aplicación del algoritmo a funciones de optimización

p_{ga}	$c = 2$			$c = 5$			$c = 10$			$c = 20$		
	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75
$Rank_{dim=5}$	8.38	8.33	6.33	5.25	4.50	5.83	6.13	5.50	5.08	8.29	6.25	5.13
$Rank_{dim=10}$	5.17	5.58	5.63	4.50	4.75	4.54	7.38	6.29	6.29	11.00	8.79	8.04
$Rank_{dim=20}$	2.63	3.29	3.92	5.17	4.42	5.46	8.63	6.58	6.88	11.67	9.88	9.50
$Rank_{dim=40}$	3.25	2.67	2.75	5.25	4.33	5.38	9.00	7.33	7.54	11.25	9.83	9.42
$Rank_{general}$	5.00	4.75	5.00	3.75	3.00	5.00	8.50	6.50	6.25	11.50	10.25	8.25

Tabla 4.16: Ranking promedio de los diferentes algoritmos utilizados por dimensión y valor de p_{ga}

tomar, podemos llegar a la conclusión de que el tamaño de la población genética se encuentra en el intervalo $GA_{tam} \in [3, 600]$ y la población a la que se aplica la CE estaría contenida en el intervalo $CE_{tam} \in [2, 600]$. Un ejemplo para entender lo expuesto: para $dim = 5$ y $c = 2$, GA_{tam} tomará los valores $GA_{tam} = \{3, 5, 8\}$ y $CE_{tam} = \{7, 5, 2\}$, siendo CE_{tam} calculada como se describió en la experimentación sobre congestión, es decir $CE_{tam} = POB_{tam} - GA_{tam}$.

Se realiza en primera instancia, por tanto, un estudio de estos dos parámetros. Para que los resultados que se han obtenido se puedan entender de una manera sencilla, la Tabla 4.16 provee los rankings promedio obtenidos para cada dimensión, y un ranking general, en el cuál no se tiene en cuenta la dimensión tratada. Se ha utilizado como condición de parada para esta experimentación un total de 25000 evaluaciones, de cara a no extender demasiado la experimentación en el tiempo y ser, a la vez, suficientes para el estudio que nos ocupa. Las funciones utilizadas han sido optimizadas entre los rangos $R_{min} = -5$ y $R_{max} = 5$, tal y como se indica en la documentación provista por la competición BBOB. Lo que buscamos con este estudio es encontrar qué tamaño de población y qué porcentaje p_{ga} aproximado es necesario para obtener los mejores resultados posibles para la mayoría de las funciones. Los valores resaltados representan los dos mejores rankings promedio en cada una de las dimensiones consideradas.

Para $dim = 5$, los mejores valores se consiguen en configuraciones con $c = \{5, 10\}$ y $p_{ga} = \{0.5, 0.75\}$. En el caso de $dim = 10$ los tres mejores valores se dan con la constante $c = 5$. Para el resto de dimensiones ($dim = \{20, 40\}$), $c = 2$ con todos los valores posibles de p_{ga} consiguen todos mejores rankings. Basándonos en estos resultados, podemos decir que, para dimensiones menores o iguales a $dim = 10$, la constante debería tomar el valor $c = 5$, mientras que, para el resto de casos, la constante debería tomar el valor $c = 2$. De una manera más formal, la fórmula para determinar el tamaño de la población se definiría como en la Ecuación 4.9.

$$POB_{tam} = \begin{cases} 5 \cdot dim & \text{si } dim \leq 10 \\ 2 \cdot dim & \text{si } dim > 10 \end{cases} \quad (4.9)$$

4. Experimentación y resultados

Dados estos valores, y habiendo tanteado el valor óptimo para el porcentaje de GA_{tam} , se realiza un estudio más exhaustivo del parámetro p_{ga} . Además, como se explicó en secciones anteriores, CE cuenta con un parámetro, n_{up} , que determina el número de individuos que utiliza para actualizar la media y la desviación previa con el fin de crear nuevos individuos. El estudio de este parámetro, junto con el porcentaje p_{ga} , resulta fundamental para poder encontrar la configuración óptima. Por tanto, al igual que p_{ga} se encarga del tamaño de la población genética GA_{tam} , se hace necesaria la creación de un porcentaje p_{up} que ayudará a determinar el valor del número de individuos n_{up} . El valor de n_{up} dependerá a su vez del tamaño de la población de CE (CE_{tam}), dado que el número de individuos utilizados para actualizar los valores de media y desviación no podrá superar el tamaño total de la población en la que se aplicará la CE.

Para este estudio, p_{up} tomará los valores $p_{up} = \{0.2, 0.4, 0.6, 0.8, 1\}$, mientras que $p_{ga} = \{0, 0.05, 0.1 \dots, 1\}$. Se estudia cada posible par (p_{ga}, p_{up}) para determinar la mejor combinación posible. Un ejemplo: $(p_{ga}, p_{up}) = (0.4, 0.2)$ denotará una ejecución en el que el 40 % de los individuos de la población POB_{tam} formarán la sub-población genética, mientras que el restante 60 % formará la sub-población dedicada a CE. De este último 60 %, el 20 % de los mejores individuos serán los utilizados para actualizar los valores de media y desviación asociados al método de entropía cruzada.

Para mostrar el resultado, se ha hecho uso de mapas de calor que denotan el rango que se obtiene para cada combinación de parámetros en cada dimensión. Cuanto más oscuro sea el cuadrado, mejor será el ranking. La Figura 4.17 muestra dichos mapas de calor mientras que la Tabla 4.17 resume los valores tomados para los parámetros en esta parte de la experimentación.

Parámetros	Valor
POB_{tam}	Ecuación 4.9
Porcentaje de inviduos del GA	$p_{ga} = \{0, 0.05, 0.1 \dots 1\}$
Nº de inviduos utilizados para actualizar	$p_{up} = \{0.2, 0.4 \dots 1\}$
Condición de parada	$T_{max} = 25000$

Tabla 4.17: Valores de los diferentes parámetros utilizados en la segunda parte de la experimentación

Con respecto a las figuras, el eje X muestra los porcentajes de GA_{tam} mientras que el eje Y muestra los valores de p_{up} utilizados. Las figuras superiores muestran los resultados para las dimensiones 5 y 10 mientras que las figuras inferiores lo hacen para las dimensiones 20 y 40. En cada figura se muestra una leyenda (colocada a la derecha) con la escala de grises y los valores a los que corresponde cada color.

4.2 Aplicación del algoritmo a funciones de optimización

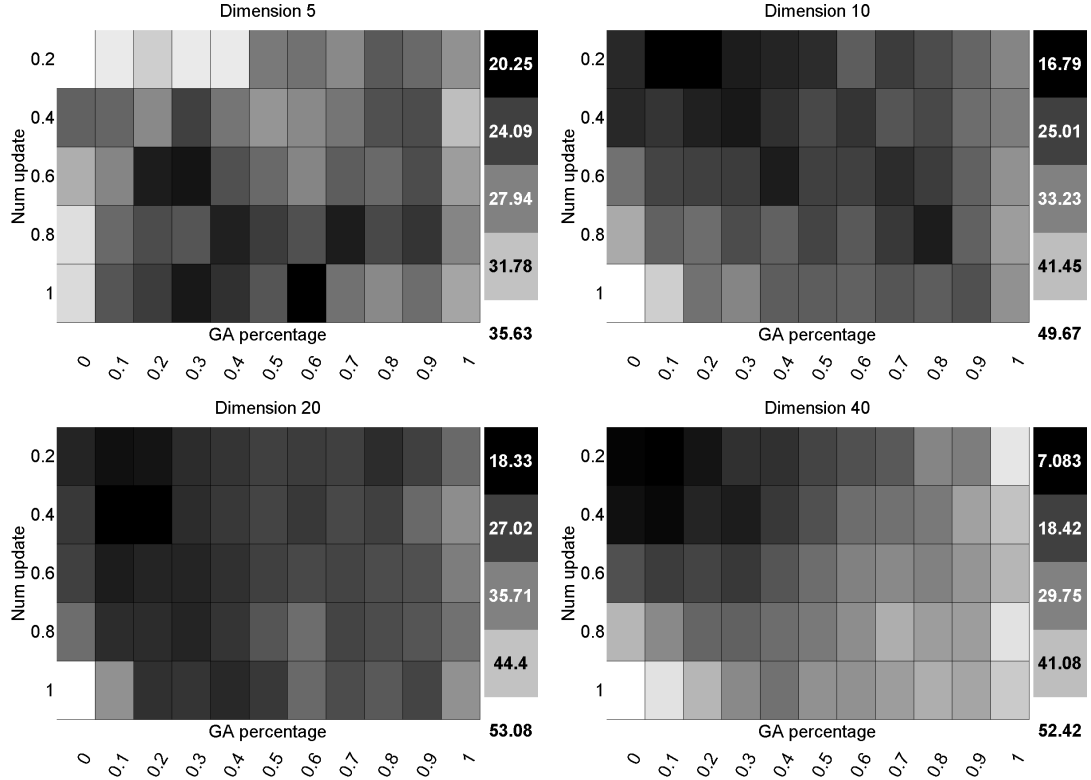


Figura 4.17: Ranking promedio de cada dupla de valores utilizada en los posibles valores de dimensión dim . El eje x contiene los porcentajes posibles de p_{ga} mientras que en el eje y se exponen los valores del porcentaje p_{up} . En la parte superior se muestran los gráficos de calor de $dim = 5$ (izquierda) y $dim = 10$ (derecha). En la parte inferior, los correspondientes a $dim = 20$ (izquierda) y $dim = 40$ (derecha).

Comenzando por la dimensión 5 ($dim = 5$), los mejores valores se encuentran en los intervalos $p_{up} \in [0.6, 1]$ y $p_{ga} \in [0.3, 0.7]$, mientras que los peores valores están situados en la fila de $p_{up} = 0.2$ y en la columna de $p_{ga} = 0$. Para el resto de dimensiones, los mejores resultados quedan concentrados en los intervalos $p_{up} \in [0.2, 0.6]$ y $p_{ga} \in [0, 0.3]$. Cuando el valor de la dimensión aumenta, es fácil ver que las mejores combinaciones de estos parámetros se obtienen con valores bajos de ambos, dado que es donde se encuentran los cuadros más oscuros. Además, cuanto mayor es la dimensión, más aumenta la diferencia entre los mejores y los peores rankings. Una de las combinaciones obtenidas que obtienen un buen rendimiento en dimensiones altas y razonablemente bueno en bajas es $p_{up} = 0.4$ y $p_{ga} = 0.1$. Se utilizarán estos valores para la comparativa entre técnicas existentes en la literatura en la siguiente sección.

4. Experimentación y resultados

Parámetros	Valor
POB_{tam}	Ecuación 4.9
Porcentaje de individuos del GA	$p_{ga} = 0.1$
Porcentaje de individuos del CE	$p_{ce} = 0.9$
Parámetros del GA	$p_c = 0.9, p_m = \frac{1}{dim}, \alpha = 0.5$
Parámetros del CE	$L_r = 0.7, p_{up} = 0.4$
Condición de parada	$T_{max} = 25000$

Tabla 4.18: Parámetros utilizados para la comparativa entre el algoritmo y los métodos del estado del arte tras los estudios realizados

4.2.3 Comparativa de los resultados

Esta sección contiene la comparativa entre el método propuesto con los parámetros elegidos durante los diferentes estudios realizados anteriormente y las técnicas existentes en la literatura. Estos parámetros y el valor que toman quedan recogidos en la Tabla 4.18.

Con la idea de evitar que tanto el algoritmo propuesto como los de la literatura usados en la comparativa siempre converjan al mismo valor, se han utilizado un total de 15 instancias, todas ellas con valores óptimos diferentes. Las técnicas usadas para la comparativa pertenecen al workshop BBOB 2013 realizado durante la conferencia GECCO del mismo año. Las técnicas incluyen un GA con codificación real, dos tipos de GA celulares, un GA generacional y un método Hill-Climber. Se han escogido los siguientes métodos para tener una comparativa de calidad, dado que se utilizaron también dichas funciones. Otro motivo es el de poder alcanzar un mejor rendimiento que técnicas como un GA como es el de codificación real, una técnica con el mismo enfoque que la propuesta, utilizando varias poblaciones (en el caso del GA celular) o una técnica realmente simple (como el Hill-Climber). Las configuraciones usadas para estos métodos son las indicadas por sus autores en sus artículos originales, los cuáles se indican junto con la descripción y las características dadas a continuación:

- ◇ GA con codificación real basado en proyección aumentada (PRCGA). Este algoritmo cuenta con cinco operadores: selección por torneo, cruce blend- α , mutación no uniforme, proyección y un mecanismo para evitar el estancamiento. La probabilidad de cruce se establece a 0.8, la de mutación a 0.15 y el tamaño del torneo a 3. Para el proceso de experimentación, las conclusiones o más detalles, consultar [Sawyer 13].
- ◇ La implementación de la Evolución Diferencial (DE) aplicada en [Pošík 12] en su forma estándar, con 'best/1' como operador de mutación.

4.2 Aplicación del algoritmo a funciones de optimización

- ◇ Los GA celulares (CGA) son una variedad de los GA paralelos. Se definen como GAs en los que la población es dividida en dos sub-poblaciones semi-separadas. En este caso, cada individuo es su propia sub-población. Dos de estos algoritmos se utilizan para la comparativa. Uno de ellos, llamado Grid, se implementa en su forma estándar tal y como se describe en [Alba 09], mientras que el otro es un “ring” CGA, desarrollado en [Holtschulte 13].
- ◇ Se utiliza un GA generacional (GGA) con selección por ranking para la comparativa.
- ◇ Por último, un algoritmo Hill-Climber (Hill) [Jacobson 04] es una técnica de optimización iterativa que pertenece a la familia de las técnicas de búsqueda local. En esta comparativa se utiliza la técnica desarrollada en [Holtschulte 13].

Las Tablas 4.19–4.22 muestran el error medio. Se ha considerado el error como el mejor fitness obtenido menos el óptimo de la instancia utilizada. Es decir $error = F_{mejor} - F_{opt}$. Como condición de parada para las técnicas utilizadas, se utiliza el mismo número de evaluaciones que se han empleado anteriormente, 25000. Los valores en negrita representan los dos mejores valores obtenidos para cada función. Los resultados con un asterisco * indican el mejor valor para el error alcanzado para cada función.

Para $dim = 5$, GACE obtiene uno de los dos resultados destacados en 11 de las 24 funciones, y el más cercano al óptimo de esos resultados en 3 de ellas. El mayor número de resultados destacados lo obtiene la técnica *DE* con 16 de 24, siendo la mejor en 14 de ellos. *PRCGA*, *Ring* y *GGA* muestran un rendimiento similar con 5, 5 y 7 valores entre los mejores resultados respectivamente. Las técnicas con peor rendimiento han sido *Grid*, que sólo obtiene el resultado superior en dos ocasiones, y *Hill*, que obtiene el mejor resultado en 4 funciones.

En el caso de $dim = 10$, *Ring* y *Grid* obtienen 1 y 4 de los resultados destacados, siendo las técnicas que menos tienen en esta dimensión. Centrándonos en *DE*, obtiene uno de los dos resultados destacados en 13 de 24 funciones, siendo la técnica que obtiene el valor más cercano al óptimo en 8 de esos casos. Por otro lado, *PRCGA* mejora su rendimiento hasta contar con 6 valores destacados entre los resultados. GACE también mejora, obteniendo 14 de los mejores resultados y siendo en 6 de ellos la que menos error ha obtenido. Esto la coloca como la segunda técnica en lo que a resultados se refiere para esta dimensión.

Para la experimentación con una dimensión $dim = 20$, las técnicas que obtienen más resultados destacados son *PRCGA* y *GACE*, con 9 y 18 entre los mejores valores respectivamente. De entre las 18 funciones en las que *GACE* destaca, obtiene el resultado más cercano al óptimo en 8 de ellas. Como se puede observar, hay una diferencia entre los resultados obtenidos en las anteriores

4. Experimentación y resultados

	GACE	DE	GGA	Grid	Hill	Ring	PRCGA
F_1	0.00e+00*	7.08e-09	1.19e-04	1.03e-04	1.58e-05	2.28e-04	2.70e-05
F_2	6.62e-06	6.13e-09*	1.70e+00	3.35e+00	2.09e+00	1.35e+00	1.81e-03
F_3	2.72e-03	2.00e-01	5.44e-02	7.20e-02	1.49e-02*	1.24e-01	4.15e-01
F_4	1.93e+00	1.89e+00	9.50e-02	2.15e-01	3.99e-02*	2.46e-01	8.67e-01
F_5	-1.02e-14*	-1.02e-14*	-1.02e-14*	-1.02e-14*	-1.02e-14*	-1.02e-14*	2.24e+00
F_6	2.48e-01	7.58e-09*	1.53e-01	4.44e-01	9.16e-02	2.80e-01	1.60e+00
F_7	6.12e-01	4.49e-09*	1.38e-01	6.98e-01	9.74e-01	2.04e-01	2.26e-01
F_8	3.54e+00	5.29e-01*	1.47e+00	1.69e+00	1.40e+00	8.25e-01	1.50e+00
F_9	3.85e+00	5.30e-01*	2.39e+00	3.78e+00	3.47e+00	1.76e+00	1.73e+00
F_{10}	5.04e+02	2.61e-02*	2.88e+02	9.87e+02	8.43e+02	2.09e+02	2.06e+03
F_{11}	2.37e+01	1.38e-03*	6.42e+00	3.54e+01	1.38e+01	8.07e+00	1.99e+01
F_{12}	7.05e+00	1.71e+00*	7.48e+01	6.38e+01	5.73e+01	3.82e+02	5.78e+00
F_{13}	3.76e+00	1.34e-03*	1.34e+01	1.15e+01	3.12e+01	8.34e+00	4.59e+00
F_{14}	5.05e-04	9.94e-08*	3.14e-03	1.00e-02	9.80e-03	5.81e-03	2.90e-03
F_{15}	3.11e+00	2.03e+00*	3.61e+00	1.12e+01	1.41e+01	5.78e+00	6.96e+00
F_{16}	3.14e-01*	9.87e-01	4.66e-01	1.01e+00	1.59e+00	6.51e-01	5.98e-01
F_{17}	6.24e-03	1.38e-04*	6.15e-02	4.27e-01	1.01e+00	1.40e-01	8.71e-02
F_{18}	1.18e-01	2.24e-02*	3.31e-01	2.44e+00	2.43e+00	6.10e-01	2.72e-01
F_{19}	1.77e-01	4.01e-01	3.83e-01	6.96e-01	6.36e-01	6.14e-01	3.04e-02*
F_{20}	5.63e-01	2.48e-01	1.94e-01	4.80e-01	5.04e-01	1.68e-01*	8.66e-01
F_{21}	2.35e+00	8.38e-01	6.50e-01	1.99e+00	2.80e+00	6.06e-04*	4.73e-01
F_{22}	1.62e+00	1.09e+00	1.04e+00	2.78e+00	1.59e+00	1.49e-01*	2.32e+00
F_{23}	9.93e-01	1.19e+00	1.00e+00	8.84e-01*	9.60e-01	9.06e-01	9.33e-01
F_{24}	9.81e+00	9.76e+00	1.11e+01	1.21e+01	1.08e+01	1.21e+01	7.71e+00*

Tabla 4.19: Media del error de las técnicas para $dim = 5$

dimensiones y la que se trata actualmente, ya que el rendimiento de *GACE* mejora dada la obtención de mejores resultados. Para este valor de dimensión, *DE* obtiene uno de los dos mejores resultados sólo en 7 de 24 funciones, siendo el mejor en 5 de ellas. En el caso de *Grid* y *Ring*, siguen obteniendo los resultados más alejados del óptimo también en esta dimensión.

Por último, para la dimensión $dim = 40$, las técnicas con mayor diferencia entre el óptimo y el valor obtenido son *GGA* y *Ring*, con 3 de 24 resultados, y *DE*, con sólo un valor entre los mejores. *Grid* sigue estando entre las técnicas que obtienen mayores diferencias con 4 resultados mientras *Hill* mejora con respecto a la dimensión anterior, con 9 de 24 posibles resultados. En este caso, la propuesta es la mejor técnica con 19 de 24 valores destacados, 12 de los cuales son el valor más cercano al óptimo, seguido de *PRCGA* con 13.

4.2 Aplicación del algoritmo a funciones de optimización

	GACE	DE	GGA	Grid	Hill	Ring	PRCGA
F_1	0.00e+00*	8.36e-09	1.82e-03	1.42e-03	1.68e-04	1.05e-02	4.21e-06
F_2	1.53e-05	8.27e-09*	1.36e+01	1.65e+01	1.30e+01	4.27e+01	1.07e-01
F_3	3.02e+00	1.12e+00	6.38e-01	1.09e+00	8.14e-02*	2.54e+00	1.54e+00
F_4	7.88e+00	3.08e+00	1.27e+00	1.74e+00	2.65e-01*	4.08e+00	2.47e+00
F_5	5.95e-13	5.86e-15*	5.86e-15*	5.86e-15*	5.86e-15*	5.86e-15*	9.52e+00
F_6	3.10e+00	1.06e-02*	1.36e+00	5.17e+00	2.49e+00	4.48e+00	1.76e+00
F_7	2.18e+00	1.82e-01*	2.33e+00	7.08e+00	6.11e+00	3.23e+00	4.08e+00
F_8	9.58e+00	2.38e+00	1.48e+01	2.23e+01	2.13e+00*	1.02e+01	1.12e+01
F_9	2.52e+01	5.53e+00	1.15e+01	7.33e+01	5.55e+01	1.27e+01	4.76e+00*
F_{10}	3.55e+03*	1.31e+04	4.47e+03	1.81e+04	1.84e+04	6.61e+03	6.40e+03
F_{11}	4.64e+01	8.39e+01	3.15e+01*	9.79e+01	6.87e+01	4.26e+01	4.00e+01
F_{12}	3.13e+00*	1.91e+02	1.28e+03	2.17e+03	1.33e+03	1.17e+04	1.71e+01
F_{13}	6.17e+00	2.38e+00*	1.73e+01	3.65e+01	1.80e+01	3.93e+01	1.15e+01
F_{14}	3.60e-03	7.49e-04*	1.61e-02	2.70e-02	1.44e-02	3.24e-02	4.54e-03
F_{15}	6.79e+00*	3.85e+01	2.27e+01	5.93e+01	7.61e+01	3.87e+01	1.28e+01
F_{16}	1.44e+00*	1.01e+01	2.40e+00	6.39e+00	6.28e+00	3.41e+00	2.43e+00
F_{17}	1.50e-02	2.98e-03*	2.89e-01	3.11e+00	3.25e+00	9.11e-01	2.75e-01
F_{18}	1.82e-01*	4.22e-01	1.32e+00	1.06e+01	1.61e+01	3.11e+00	1.09e+00
F_{19}	1.27e+00	2.81e+00	2.37e+00	3.46e+00	3.21e+00	3.00e+00	3.30e-02*
F_{20}	1.29e+00	3.55e-01*	5.16e-01	7.17e-01	6.65e-01	7.13e-01	1.30e+00
F_{21}	3.74e+00	2.77e+00	1.92e+00	6.03e+00	1.48e+01	4.36e-01*	4.01e+00
F_{22}	7.80e+00	2.21e+00	4.22e+00	1.47e+01	6.59e+00	7.51e-01*	4.79e+00
F_{23}	1.66e+00	1.90e+00	1.41e+00	1.47e+00	1.32e+00*	1.38e+00	1.45e+00
F_{24}	3.56e+01	5.03e+01	4.67e+01	6.39e+01	5.59e+01	5.56e+01	2.93e+01*

Tabla 4.20: Error medio de las técnicas para $dim = 10$

Para resumir todos estos resultados, en un total de 96 casos de estudio (24 funciones por 4 dimensiones utilizadas), *GACE* obtiene uno de los dos mejores valores en 62 casos, de los cuales alcanza el menor error en 29 de ellos. La segunda mejor técnica ha sido *DE*, con un resultado entre los mejores en 37 de los 96 casos, siendo el mejor en 28 de ellos. Para asegurarnos de que las diferencias que se han encontrado en lo que a rendimiento se refiere son o no significativas, se hace necesario el uso de test estadísticos. Para ello, se han utilizado dos test estadísticos siguiendo las indicaciones propuestas en [Derrac 11].

En primer lugar, se aplica el test de Friedman [Derrac 11] para conocer si hay diferencias significativas entre los métodos comparados. La Tabla 4.23 muestra los rankings proporcionados por este test no paramétrico para cada uno de los algoritmos en cada dimensión y globalmente sobre

4. Experimentación y resultados

	GACE	DE	GGA	Grid	Hill	Ring	PRCGA
F_1	5.13e-05	8.62e-06*	2.65e-02	2.12e-02	1.33e-03	2.56e-01	1.82e-04
F_2	1.13e-02*	3.94e-02	1.10e+02	2.23e+02	6.54e+01	8.76e+02	3.16e+01
F_3	9.87e+00	9.45e+01	5.71e+00	7.61e+00	8.55e-01*	2.33e+01	6.20e+00
F_4	2.63e+01	9.07e+01	9.90e+00	1.09e+01	1.53e+00*	3.34e+01	1.20e+01
F_5	2.85e-09	-3.61e-15*	6.46e-14	6.46e-14	6.46e-14	6.46e-14	1.19e+01
F_6	1.47e+01*	1.80e+02	1.49e+01	4.60e+01	2.66e+01	6.66e+01	2.57e+01
F_7	1.32e+01	2.13e+01	1.22e+01*	4.14e+01	3.38e+01	2.79e+01	1.72e+01
F_8	2.64e+01	2.59e+01*	7.72e+01	9.49e+01	4.80e+01	8.26e+01	6.46e+01
F_9	3.66e+01	2.81e+01*	5.29e+01	9.44e+01	7.62e+01	1.01e+02	4.03e+01
F_{10}	3.54e+04	3.27e+05	4.26e+04	8.23e+04	4.28e+04	4.68e+04	3.11e+04*
F_{11}	8.86e+01*	2.46e+02	9.76e+01	2.04e+02	2.21e+02	1.51e+02	9.04e+01
F_{12}	3.13e+02	3.89e+02	2.46e+04	1.90e+04	1.42e+03	2.60e+05	1.10e+02*
F_{13}	2.16e+01	1.27e+01*	6.67e+01	6.10e+01	2.46e+01	1.49e+02	2.19e+01
F_{14}	1.31e-02*	2.33e-02	5.25e-02	6.40e-02	2.65e-02	3.05e-01	1.88e-02
F_{15}	2.11e+01*	1.47e+02	9.16e+01	2.29e+02	2.61e+02	1.48e+02	4.26e+01
F_{16}	2.53e+00*	2.88e+01	7.73e+00	1.36e+01	1.35e+01	1.09e+01	5.45e+00
F_{17}	1.75e-01*	5.70e-01	6.69e-01	8.13e+00	1.12e+01	3.29e+00	7.40e-01
F_{18}	8.47e-01*	5.39e+00	4.76e+00	2.91e+01	4.24e+01	1.20e+01	2.70e+00
F_{19}	2.90e+00	6.21e+00	5.24e+00	7.78e+00	7.35e+00	6.74e+00	2.06e-01*
F_{20}	2.29e+00	2.66e+00	9.07e-01	1.17e+00	8.82e-01*	1.44e+00	1.63e+00
F_{21}	4.61e+00	5.73e+00	5.01e+00	1.20e+01	1.86e+01	1.54e+00*	1.12e+01
F_{22}	9.15e+00	8.80e+00	7.55e+00	2.38e+01	1.31e+01	3.33e+00*	7.93e+00
F_{23}	2.43e+00	3.45e+00	2.66e+00	2.42e+00	2.10e+00*	2.51e+00	2.42e+00
F_{24}	1.45e+02	1.60e+02	1.51e+02	2.38e+02	2.51e+02	2.07e+02	1.11e+02*

Tabla 4.21: Error medio de las técnicas para $dim = 20$

todas las dimensiones.

Como se puede ver en dicha tabla, el algoritmo propuesto obtiene el mejor ranking en $dim = 20$ y $dim = 40$, y en términos globales. Se puede apreciar como la diferencia entre el ranking obtenido por nuestra técnica y los diferentes algoritmos con los que se ha comparado es mayor cuando tanto las dimensiones como, por tanto, la complejidad del problema aumentan.

Los test post-hoc de Holm [Holm 79] y Finner [Finner 93] se aplican a los resultados obtenidos por el test de Friedman usando DE como método de control en $dim = \{5, 10\}$ y $GACE$ en el resto de dimensiones. La Tabla 4.24 muestra los p-valores ajustados devueltos por dichos tests para cada dimensión y técnica. Los valores que se ofrecen están redondeados a tres decimales. Para resaltar las diferencias significativas, los p-valores por debajo de 0.1 se muestran en negrita.

4.2 Aplicación del algoritmo a funciones de optimización

	GACE	DE	GGA	Grid	Hill	Ring	PRCGA
F_1	3.98e-08*	7.33e+00	4.25e-01	2.62e-01	1.17e-02	3.93e+00	7.27e-03
F_2	8.52e+00*	3.12e+04	2.26e+03	2.71e+03	6.74e+02	2.35e+04	8.04e+01
F_3	4.31e+01	3.98e+02	4.21e+01	3.99e+01	7.13e+00*	1.43e+02	4.51e+01
F_4	6.65e+01	6.83e+02	5.77e+01	6.27e+01	1.12e+01*	2.05e+02	7.83e+01
F_5	2.46e-01	1.34e-14*	1.34e-14*	1.34e-14*	1.34e-14*	1.34e-14*	3.99e+01
F_6	6.77e+01*	3.44e+03	1.35e+02	4.42e+02	1.56e+02	3.95e+02	1.06e+02
F_7	3.97e+01*	4.92e+02	9.57e+01	1.70e+02	1.57e+02	1.56e+02	6.61e+01
F_8	5.44e+01*	3.39e+03	2.59e+02	2.45e+02	1.06e+02	8.31e+02	1.82e+02
F_9	4.28e+01*	2.88e+03	2.48e+02	4.33e+02	9.38e+01	1.04e+03	1.89e+02
F_{10}	1.12e+05*	2.94e+06	1.76e+05	3.73e+05	1.54e+05	4.18e+05	1.24e+05
F_{11}	1.90e+02*	7.06e+02	2.91e+02	4.86e+02	4.56e+02	3.51e+02	1.99e+02
F_{12}	5.44e+02*	2.44e+07	4.22e+05	3.36e+05	3.29e+04	5.40e+06	8.63e+03
F_{13}	7.06e+01	9.69e+02	2.25e+02	1.90e+02	5.79e+01*	5.89e+02	9.96e+01
F_{14}	2.65e-02*	1.91e+01	3.40e-01	2.54e-01	5.57e-02	4.02e+00	8.27e-02
F_{15}	8.99e+01*	6.04e+02	2.96e+02	7.58e+02	8.44e+02	5.81e+02	1.47e+02
F_{16}	1.05e+01	4.48e+01	1.52e+01	2.67e+01	2.40e+01	2.29e+01	9.34e+00*
F_{17}	1.99e-01	7.75e+00	1.79e+00	1.34e+01	2.11e+01	7.84e+00	1.49e+00*
F_{18}	1.01e+00*	3.01e+01	8.69e+00	5.14e+01	7.92e+01	2.73e+01	5.98e+00
F_{19}	5.90e+00	9.50e+00	8.02e+00	1.29e+01	1.16e+01	9.68e+00	2.50e-01*
F_{20}	3.14e+00	8.55e+01	1.57e+00	1.52e+00	8.77e-01*	2.89e+00	1.81e+00
F_{21}	5.98e+00	4.82e+01	6.35e+00	7.15e+00	1.69e+01	5.27e+00	4.95e+00*
F_{22}	7.83e+00	3.91e+01	9.00e+00	2.30e+01	1.74e+01	5.43e+00*	9.49e+00
F_{23}	4.06e+00	5.97e+00	4.24e+00	3.33e+00	2.99e+00*	3.66e+00	4.39e+00
F_{24}	3.87e+02	6.36e+02	3.80e+02*	7.68e+02	8.19e+02	6.18e+02	4.42e+02

Tabla 4.22: Error medio de las técnicas para $dim = 40$

De manera general, *GACE* es significativamente mejor que el resto de métodos excepto uno, *PRCGA*. En este caso, aunque la diferencia no es significativa, el nivel de confianza es alto, concretamente un 88 %.

Si nos fijamos en los resultados obtenidos en cada dimensión, para $dim = 5$, *DE* mejora significativamente a *Grid*, *Hill* y *Ring*. En $dim = 10$, *DE* mejora significativamente a *Grid*, *Hill* y *Ring* según Finner, mientras que, según Holm, sólo mejora a *Grid* y *Ring*. Finalmente, para las dimensiones restantes ($dim = 20, 40$), ambos tests indican que *GACE* mejora significativamente a todos los métodos menos a *PRCGA*. Concretamente, en la dimensión $dim = 40$, se puede encontrar el valor más cercano entre los métodos *PRCGA* y *GACE*.

Con los resultados aportados por los experimentos realizados a lo largo de esta sección, podemos

4. Experimentación y resultados

	GACE	DE	GGA	Grid	Hill	Ring	PRCGA
$Rank_{dim=5}$	3.69	2.44	3.60	5.5	4.81	3.83	4.12
$Rank_{dim=10}$	3.19	3.02	3.23	5.85	4.60	4.69	3.42
$Rank_{dim=20}$	2.46	4.21	3.5	5.35	4.46	5.12	2.89
$Rank_{dim=40}$	2.08	6.33	3.54	4.71	3.75	4.71	2.87
$Rank_{all}$	2.85	4	3.47	5.35	4.41	4.59	3.33

Tabla 4.23: Resultados del test de Friedman para cada dimensión

	$dim = 5$		$dim = 10$			$dim = 20$		$dim = 40$		Global	
	Holm	Finner	Holm	Finner		Holm	Finner	Holm	Finner	Holm	Finner
GACE	0.09	0.054	1.577	0.799	DE	0.015	0.007	0	0	0.001	0
GGA	0.09	0.061	1.577	0.799	GGA	0.189	0.113	0.039	0.023	0.097	0.058
Grid	0	0	0	0	Grid	0	0	0	0	0	0
Hill	0.001	0.001	0.062	0.022	Hill	0.005	0.003	0.022	0.011	0	0
Ring	0.076	0.037	0.038	0.022	Ring	0	0	0	0	0	0
PRCGA	0.027	0.013	1.577	0.673	PRCGA	0.483	0.483	0.204	0.204	0.128	0.128

Tabla 4.24: Estadísticas de los test de Holm y Finner para cada dimensión y técnica

llegar a las siguientes conclusiones:

- ◇ Comenzando por el tamaño de la población general (POB_{tam}), se ha definido este con valores, en general, bajos, dado que va desde 25 individuos en el caso de $dim = 5$ hasta 80 en el caso de $dim = 40$, siendo en este último donde se dan los resultados más óptimos.
- ◇ Con respecto a los porcentajes que determinan la población genética utilizada (p_{ga}) y la actualización de la media y desviación necesarias para CE (p_{up}), un valor pequeño del primero, entre el 0 y el 30 %, y uno intermedio en el caso del segundo, entre 20 % y 60 % de la población de CE, ofrecen un rendimiento óptimo tanto en dimensiones bajas como en altas.
- ◇ Independientemente del tipo de función, *GACE* obtiene un gran rendimiento en dimensiones altas, dejando atrás a técnicas del estado del arte en optimización tan conocidas como *DE*.
- ◇ Sin embargo, en dimensiones bajas ($dim = \{5, 10\}$), aunque su rendimiento se asemeja al de *DE*, es superado por esta técnica.
- ◇ En general, y para todas las dimensiones utilizadas, *GACE* sería la técnica elegida dada su posición en el ranking promedio de resultados.
- ◇ Estadísticamente, supera en dimensiones altas a aquellas técnicas con las que se le ha comparado.

*Que el que llore el último llore peor. Y que
los últimos no sean los de siempre.*

Marta Martínez Carro

5

Conclusiones y trabajo futuro

Este capítulo contiene las conclusiones a las que se ha llegado tras los resultados obtenidos durante la experimentación llevada a cabo en esta tesis. En primer lugar, la Sección 5.1 muestra la difusión de los resultados extraídos durante la realización de este proyecto en las diferentes revistas de impacto y congresos, tanto nacionales como internacionales. La Sección 5.2 contiene las reflexiones que se han extraído de la aplicación del algoritmo propuesto a los sistemas difusos para la predicción de congestión. Tras esto, en la Sección 5.3 se exponen las conclusiones detrás del estudio de los parámetros usando GACE en funciones benchmark. Por último, las respuestas a la hipótesis planteada al comienzo de esta memoria y las líneas de trabajo que se seguirán tras la finalización de esta memoria se encuentran en la Sección 5.4.

5.1 Difusión de los resultados

Durante los años en los que se ha realizado el proyecto que conlleva como colofón final la escritura de este documento, los resultados que se han ido obteniendo con las diferentes experimentaciones y problemas se han ido difundiendo en diferentes congresos, a nivel nacional e internacional, y revistas con alto índice de impacto. A continuación, se listan las publicaciones derivadas del trabajo de tesis aquí presentado:

5. Conclusiones y trabajo futuro

- ◇ **P. Lopez-Garcia**, E. Onieva, E. Osaba, A. Masegosa, A. Perallos, *A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross-entropy*, IEEE Transactions Intelligent Transportation Systems, 2016, Volumen 17, Issue 2, Páginas 557-569. Esta revista cuenta con un factor de impacto de 2.377 en el Journal Citations Report del año 2014, y se encuentra indexada en las siguientes posiciones:
 1. Posición 13 de 125 de la categoría Civil Engineering, cuartil Q1.
 2. Posición 41 de 249 en la categoría Electrical & Electronic Engineering, cuartil Q1.
 3. Posición 9 de 33 de la categoría Transportation Science & Technology, cuartil Q2.
- ◇ **P. Lopez-Garcia**, E. Onieva, E. Osaba, A.D. Masegosa, A. Perallos, *GACE: A meta-heuristic based in the hybridization of Genetic Algorithms and Cross Entropy methods for continuous optimization*, Expert Systems with Applications. 2016, Volumen 55, páginas 508–519. Esta revista cuenta con un factor de impacto de 2.240 en el Journal Citations Report del año 2014, así como ocupa las siguientes posiciones en diferentes categorías:
 1. Posición 29 de 123 de la categoría Computer Science, Artificial Intelligence, cuartil Q1.
 2. Posición 48 de 249 de la categoría Electrical & Electronic Engineering, cuartil Q1.
 3. Posición 12 de 81 de la categoría Operations Research & Management Science, cuartil Q1.
- ◇ **P. Lopez-Garcia**, E. Onieva, A. Perallos, *Optimización de Sistemas Basados en Reglas Difusas para la predicción de congestión a corto plazo*, Jornadas científico-técnicas y seminario doctoral SEMATICA 2014.
- ◇ **P. Lopez-Garcia**, E. Onieva, A. Perallos, L. Arjona, E. Osaba, *Optimización de Sistemas Basados en Reglas Difusas para la predicción de congestión a corto plazo*, Congreso Nacional sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB), 2015, en los Proceedings, páginas 621–628.
- ◇ **P. Lopez-Garcia**, E. Onieva, E. Osaba, A. Masegosa, A. Perallos, *Hybridizing Genetic Algorithm with Cross Entropy for Solving Continuous Functions*, Genetic and Evolutionary Computation Conference, 2015, en los Proceedings, páginas 763–764.
- ◇ **P. Lopez-Garcia**, M. Wozniak, E. Onieva, A. Perallos, *Hybrid Optimization Method Applied to Adaptive Splitting and Selection Algorithm*, Hybrid Artificial Intelligent Systems, HAIS 2016, en los Proceedings, páginas 742–750.

5.2 Conclusiones de la aplicación a la predicción de congestión

También, y para quien sea de interés, se añade una lista con los diferentes artículos en los que ha participado el doctorando como co-autor a lo largo del trabajo realizado:

- ◇ A. D. Masegosa, A. Bahillo, E. Onieva, **P. Lopez-Garcia**, A. Perallos, *A new optimization approach for indoor location based on Differential Evolution*, International Fuzzy Systems Association (IFSA) 2015.
- ◇ E. Osaba, F. Diaz, E. Onieva, **P. Lopez-Garcia**, R. Carballedo, A. Perallos, *A Parallel Meta-Heuristic for solving Multiple Asymmetric Traveling Salesman Problem with Simultaneous Pickup and Delivery to solve a Demand Responsive Transport Problem*, Hybrid Artificial Intelligence Systems International Conference, HAIS 2015.
- ◇ E. Osaba, E. Onieva, F. Diaz, R. Carballedo, **P. Lopez-Garcia**, A. Perallos, *A migration strategy for distributed evolutionary algorithms based on stopping non-promising subpopulations: A case study on routing problems*, International Journal of Artificial Intelligence.
- ◇ E. Osaba, X.S. Yang, F. Diaz, **P. Lopez-Garcia**, R. Carballedo, *An Improved Discrete Bat Algorithm for Symmetric and Asymmetric Traveling Salesman Problems*, Engineering Applications of Artificial Intelligence. Esta revista cuenta con un factor de impacto de 2.207 en el Journal Citations Report de 2014.

5.2 Conclusiones de la aplicación a la predicción de congestión

Como se ha mencionado a lo largo de esta memoria, uno de los objetivos tras la creación de GACE era usarlo como herramienta de optimización de las diferentes partes de los sistemas que forman una jerarquía de FRBS, es decir, variables, etiquetas y reglas. Dicha optimización es llevada a cabo con el objetivo de predecir la congestión en un punto o segmento de una carretera. Para poder realizarlo, se han creado 12 datasets con datos de una autopista en California. La predicción se ha realizado a corto plazo, en intervalos de 5, 15 y 30 minutos.

La experimentación se centraba en los diferentes tamaños de las sub-poblaciones en las que se aplicaban GA y CE respectivamente. Además, se han utilizado diferentes tipos de conjuntos de datos de manera que se pudiera probar el rendimiento del algoritmo y las jerarquías de sistemas. Se han utilizado diferentes métricas de error para valorar dicho rendimiento. Con la primera, MAE, se obtienen mejores resultados cuando GACE utiliza un mayor tamaño en la población de GA que aquellas configuraciones con el mismo o más tamaño de población CE. Estas configuraciones obtienen, además, mejores resultados que la aplicación de GA y CE en solitario.

5. Conclusiones y trabajo futuro

Se hace necesario el uso de otra métrica debido a que MAE no tiene en cuenta las diferencias entre el tipo de congestión obtenida y la esperada, por lo que se usa sMAPE para evitar este problema. Con los resultados obtenidos con esta métrica se confirma lo obtenido con MAE: configuraciones con tamaño mayor de población de GA con respecto al tamaño de la población de CE obtienen mejores resultados. Usando esta última métrica, se compara GACE con tres métodos de la literatura. Los resultados obtenidos en dicha comparación arrojan que el método propuesto es competitivo, y puede mejorar su rendimiento en varios de los datasets propuestos.

Sin tener en cuenta el tipo de dataset utilizado, se ha realizado una comparativa entre configuraciones para conocer cuál rinde mejor en qué tipo de congestión. Los resultados muestran como configuraciones altas son las mejores opciones para ayudar a los sistemas a predecir congestión de tipo Severa y Nula. En el caso de configuraciones con tamaño de población genética bajo, éstas son las mejores para congestión de tipo Ligeras. En el caso de las configuraciones puras, GA obtiene la cuarta posición como algoritmo elegido, mientras que CE se emplaza en última posición.

Teniendo en cuenta el tipo de dataset y el valor de congestión obtenido, GACE es la técnica a utilizar en 32 de los 48 casos de estudio. Las configuraciones del algoritmo son las mejores para optimizar los sistemas cuando la congestión a tratar es Nula y Severa, mientras que reduce su rendimiento a la mitad para predecir congestiones de tipo Ligeras y Moderada. A pesar del mal rendimiento obtenido, tanto GA como CE son la técnica elegida para optimizar los sistemas en 8 de los 48 casos de estudio.

Por último, centrándonos en el tipo de dataset y el horizonte de predicción, se obtienen los mejores resultados utilizando los datasets de tipo Punto y un horizonte de predicción de 5 minutos. En el caso de los datasets Sector, se obtiene lo opuesto: los mejores errores se obtienen cuando el horizonte de predicción es de 30 minutos.

5.3 Conclusiones de la aplicación a funciones de optimización

La segunda parte de la experimentación tenía el objetivo de probar el rendimiento del algoritmo propuesto en diferentes tipos de funciones de optimización. Con dicho objetivo en mente, se han utilizado un total de 24 funciones libres de ruido tomadas de una plataforma utilizada en diferentes conferencias internacionales de este ámbito.

Lo más importante en esta parte de la experimentación era elegir el valor correcto para diferentes parámetros importantes como el tamaño de la población y sub-poblaciones, el ratio de aprendizaje para la parte CE, el número de individuos necesarios para actualizar la media y desviación, etc. No sólo es importante para el desarrollo de este problema, sino de cara a futuros trabajos en

otras temáticas totalmente diferentes. Aunque es importante la modificación de dichos parámetros dependiendo del problema, realizando esta experimentación se podrán ofrecer unos valores por defecto que asegurarán el buen rendimiento del algoritmo cualquiera que sea el problema a tratar.

En primer lugar, el tamaño de la población se calcula utilizando una constante y la dimensión del problema. En el caso de dimensiones bajas, dicha constante toma el valor de $c = 5$, mientras que para dimensiones altas, esta constante toma el valor $c = 2$. Tras esto, se han utilizado gráficos de calor para delimitar el porcentaje perteneciente al tamaño de la población genética y, por otro lado, el porcentaje de individuos necesario para actualizar tanto media como desviación en la parte CE. Los resultados arrojan que una combinación de un término medio en el primer porcentaje y un valor alto en el segundo obtienen buenos resultados en dimensiones bajas, mientras que para dimensiones altas, se recomienda un valor bajo para la población genética y un valor medio para el porcentaje de individuos.

Resumiendo los resultados obtenidos, el rendimiento de la técnica propuesta aumenta conforme lo hace la dimensión de la función. Para la dimensión más baja utilizada, GACE obtiene uno de los dos mejores resultados en un total del 46 % de las funciones, mientras que prácticamente dobla dicho porcentaje en la dimensión más alta utilizada, con uno de los dos mejores valores en el 79 % de las funciones. Para el total de los 96 casos de estudio, la propuesta obtiene uno de los dos mejores resultados en 62 de los casos, mientras que obtiene el mejor de ellos en 29 de estos 62. La segunda técnica que obtiene mejores resultados, en este caso DE, alcanza uno de los dos mejores resultados en 37 de 96, y el mejor resultado en 28 de estos 37.

Se han utilizado test estadísticos como Friedman, Holm y Finner para cerciorarse de los resultados obtenidos. Utilizando DE como algoritmo de control en dimensiones bajas, y GACE para el resto de dimensiones, los resultados obtenidos por estos test prueban que GACE es significativamente diferente que todos los algoritmos utilizados excepto uno.

5.4 Mejoras y líneas futuras de trabajo

A lo largo de este capítulo, se han remarcado las conclusiones extraídas de las diferentes experimentaciones realizadas a lo largo de esta tesis. El algoritmo y los FRBS han obtenido un buen rendimiento en lo que a predicción de congestión se refiere, independientemente del tipo de dataset utilizado, pero especialmente en los datasets Punto y Sector. Tras esto, y una vez elegidos los parámetros apropiados para su uso, el algoritmo puede alcanzar un valor más cercano al óptimo en funciones matemáticas de optimización que varios conocidos algoritmos en la literatura. Por lo tanto, podemos concluir que las diferentes partes de la hipótesis expuesta en el capítulo de Introducción han sido cumplidas.

5. Conclusiones y trabajo futuro

Por una parte, la solución propuesta en este trabajo para predicción de congestión obtiene un buen rendimiento en los diferentes tipos de congestión expuestos. Con esto, se cumple la primera parte de la hipótesis en la que el algoritmo híbrido optimiza sistemas jerárquicos con la intención de predecir congestión con buenos resultados en dicha tarea. Por otro lado, la segunda parte de la hipótesis es cubierta en el momento en el que la propuesta obtiene y mejora los resultados obtenidos por otras técnicas de la literatura en lo que a optimización de funciones se refiere.

Aún así, el algoritmo tiene mucho recorrido por delante con mejoras en su rendimiento en diferentes temas. Estas mejoras pueden ser:

- ◇ Tener en cuenta los parámetros como valores a optimizar a lo largo de la ejecución del algoritmo. Ésto convierte el problema en uno multiobjetivo: es necesario no sólo optimizar la función objetivo, sino los diferentes valores de los parámetros necesarios para ofrecer un buen rendimiento.
- ◇ Diversidad de la población. Debido a que es posible encontrar los mismos individuos en ambas sub-poblaciones, a veces la diversidad de la población puede verse comprometida a lo largo del proceso. Posibles soluciones a este problema pueden ser incrementar la probabilidad de mutación a lo largo de las generaciones en la parte genética, o realizar reinicios en la población dedicada al CE para la actualización de las medias y desviaciones.

En el caso de la predicción de congestión, principal objetivo de esta tesis y donde la propuesta se ha aplicado con más detalle debido a su importancia en problemas de la vida real, el algoritmo y el diseño de la jerarquía puede ser mejorado en los siguientes aspectos:

- ◇ Mejoras en los resultados obtenidos para congestiones Ligera y Moderada. Como se ha podido ver en el capítulo Experimentación, algunas configuraciones del algoritmo propuesto no mejoran el rendimiento de los sistemas para realizar una buena predicción en estos tipos de congestión. Esto puede deberse a las similitudes entre el cálculo de ambos valores. En trabajos futuros, estos cálculos pueden cambiar de manera de se obtenga una mejor clasificación de este tipo de instancias.
- ◇ Usar otro tipo de jerarquía. Las jerarquías en serie se han utilizado en trabajos previos a esta tesis. En el caso que nos ocupa, la jerarquía en paralelo se ha aplicado y mejorado los resultados obtenidos en otros trabajos, no sólo por el uso de este tipo de jerarquía, sino por el uso del algoritmo híbrido para optimizar los sistemas. Puede ser interesante ver los resultados que se obtendrían utilizando un tipo de jerarquía híbrida en comparación con la actual.

- ◇ Datos de otras fuentes. Los diferentes conjuntos de datos utilizados en esta tesis y el artículo asociado están al alcance de cualquiera que quiera usarlos. No hay muchos conjuntos de datos de tráfico que se puedan obtener de manera gratuita. Aún así, plataformas gubernamentales como la utilizada (PeMS) permiten a los investigadores trabajar con los datos recogidos por ellos. El uso de otros datasets puede certificar el buen rendimiento de la técnica y los sistemas, y puede mejorar la calidad y el conjunto de los datasets utilizados.
- ◇ Horizontes de tiempo. En esta parte, se ha querido predecir congestión a corto plazo, por lo que los horizontes de tiempo utilizados han sido, como el propio nombre indica, pequeños, siendo estos 5, 15 y 30 minutos. Incrementar el número de horizontes, por ejemplo, de 5 en 5 hasta un total de 60 minutos pueden llegar a predecir con mayor precisión la congestión en la vía.

En lo que a funciones de optimización se refiere, aunque no era la principal tarea de la técnica propuesta, ha obtenido buenos resultados en diferentes tipos de funciones, algunas de ellas con una alta complejidad. Además, ha mejorado los resultados obtenidos por algoritmos muy conocidos de la literatura en esta temática como la Evolución Diferencial. Las posibles líneas de acción para mejorar el algoritmo en esta temática son las siguientes:

- ◇ Durante la experimentación, se han utilizado valores para los parámetros de manera que obtuviera mejores resultados en las dimensiones altas mientras mantenía un cierto nivel de calidad en dimensiones bajas. El objetivo ahora es adaptar estos parámetros a la dimensión, de manera que se ha hecho, en este caso, con el tamaño de población.
- ◇ El uso de otros tipos de funciones, así cómo mejorar el rendimiento añadiendo o adaptando algunas características del método híbrido a las funciones y no sólo los operadores puede llegar a mejorar el rendimiento, a la vez que ofrecer una experimentación más rica.

Una de las principales características de la técnica propuesta es su adaptación a los diferentes problemas cambiando sus operadores y la codificación del cromosoma. Gracias a esto, la cantidad de problemas que la técnica puede resolver es alta. Uno de los problemas donde GACE se puede aplicar es la clasificación utilizando ensembles. Los ensembles son conjuntos de clasificadores individuales que ofrecen una aproximación diferente a los problemas de clasificación utilizando clustering para el espacio de búsqueda y aplicando el mejor de los clasificadores a cada una de las áreas delimitadas. En este aspecto, GACE puede mejorar los diferentes centroides de las áreas y los pesos de los

5. Conclusiones y trabajo futuro

clasificadores en dichas áreas de manera que se mejore el resultado obtenido por los clasificadores en cada área de estudio.

Otra posible aproximación, totalmente diferente a los temas expuestos durante esta memoria, son los sistemas RFID y su problema de colisión de tags. A grandes rasgos, debido la naturaleza del canal de red, estos sistemas son propensos a colisiones en su transmisión mediante tags, lo que ocurre cuando dos o más tags transmiten diferente información simultáneamente. La Lógica Difusa se ha utilizado ya en este tema en [Arjona 16]. Siguiendo en esta línea, es un buen y diferente tema que tratar en el que GACE puede ayudar a encontrar los valores correctos para los diferentes parámetros utilizados por la Lógica Difusa y obtener, de esta manera, mejor rendimiento en esta tarea.

Siempre habrá problemas que resolver. Los problemas de la vida real como son la predicción de congestión, la predicción de la polución o los problemas de rutas son recurrentes en muchos y diferentes proyectos, como los recogidos en Horizon 2020 debido a su importancia en el día a día. Por tanto, una de las metas más importantes que alcanzar como investigador es encontrar posibles soluciones a estos problemas. Con este objetivo en mente, la mejora y desarrollo de nuevos algoritmos que puedan lidiar con grandes cantidades de datos, y ayudar a tomar decisiones sobre temas importantes como los mencionados anteriormente, son posibles caminos a seguir para presentar dichas soluciones. El método propuesto en esta tesis intenta englobar estos propósitos, y mejorar los resultados obtenidos anteriormente por otras técnicas. Por otro lado, el desarrollo de la propuesta demuestra que no todo está creado, y que, para cada problema, hay una o más soluciones posibles. alguna de ellas, mejores que otras; otras siquiera se han pensado aún. Esta memoria y todos los resultados que se muestran en ella da buena prueba de ello.



Conclusions and Future Work

This chapter contains the conclusions from the results obtained in the experimentation studies carried out along this dissertation. First, Section A.1 shows the articles and conference papers published along the fulfillment of this dissertation. Section A.2 contains the reflections that can be extracted from the results obtained by applying the proposal to congestion forecasting. After that, in Section A.3, the conclusions about the different parameter studies made by using GACE in benchmark functions are exposed. Finally, the answer to the hypothesis made at the beginning of this document of dissertation, and future work lines are contained in Section A.4.

A.1 Dissemination of the results

During the development of this project, the results obtained by the application of the technique to different problems have been presented and published in several international and national conferences, and high impact factor journals in the mentioned themes. Therefore, a list of publications with the thesis work is presented:

- ◇ **P. Lopez-Garcia**, E. Onieva, E. Osaba, A. Masegosa, A. Perallos, *A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross-entropy*, IEEE Transactions Intelligent Transportation Systems, 2016, Volumen 17, Issue 2, pages 557-569.

A. Conclusions and Future Work

This journal has an impact factor of 2.377 in 2014 Journal Citations Report, and it can be found indexed in the next positions of the different themes:

1. Position 13 out of 125 in Civil Engineering, quartile Q1.
2. Position 41 out of 249 in Electrical & Electronic Engineering, quartile Q1.
3. Position 9 out of 33 in Transportation Science & Technology, quartile Q2.

◇ **P. Lopez-Garcia**, E. Onieva, E. Osaba, A.D. Masegosa, A. Perallos, *GACE: A meta-heuristic based in the hybridization of Genetic Algorithms and Cross Entropy methods for continuous optimization*, Expert Systems with Applications. 2016, Volumen 55, pages 508–519. his journal has an impact factor of 2.240 in 2014 Journal Citations Report, and it can be found indexed in the next positions of the different themes:

1. Position 29 out of 123 in Computer Science, Artificial Intelligence, quartile Q1.
2. Position 48 out of 249 in Electrical & Electronic Engineering, quartile Q1.
3. Position 12 out of 81 in Operations Research & Management Science, quartile Q1.

◇ **P. Lopez-Garcia**, E. Onieva, A. Perallos, *Optimización de Sistemas Basados en Reglas Difusas para la predicción de congestión a corto plazo*, Jornadas científico-técnicas y seminario doctoral SEMATICA 2014.

◇ **P. Lopez-Garcia**, E. Onieva, A. Perallos, L. Arjona, E. Osaba, *Optimización de Sistemas Basados en Reglas Difusas para la predicción de congestión a corto plazo*, Congreso Nacional sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB), 2015, in Proceedings, pages 621–628.

◇ **P. Lopez-Garcia**, E. Onieva, E. Osaba, A. Masegosa, A. Perallos, *Hybridizing Genetic Algorithm with Cross Entropy for Solving Continuous Functions*, Genetic and Evolutionary Computation Conference, 2015, in Proceedings, pages 763–764.

◇ **P. Lopez-Garcia**, M. Wozniak, E. Onieva, A. Perallos, *Hybrid Optimization Method Applied to Adaptive Splitting and Selection Algorithm*, Hybrid Artificial Intelligent Systems, HAIS 2016, in Proceedings, pages 742–750.

Besides, and for whoever interested, a list of the different articles the doctoral student has participated in as co-author is presented:

- ◇ A. D. Masegosa, A. Bahillo, E. Onieva, **P. Lopez-Garcia**, A. Perallos, *A new optimization approach for indoor location based on Differential Evolution*, International Fuzzy Systems Association (IFSA) 2015.
- ◇ E. Osaba, F. Diaz, E. Onieva, **P. Lopez-Garcia**, R. Carballedo, A. Perallos, *A Parallel Meta-Heuristic for solving Multiple Asymmetric Traveling Salesman Problem with Simultaneous Pickup and Delivery to solve a Demand Responsive Transport Problem*, Hybrid Artificial Intelligence Systems International Conference, HAIS 2015.
- ◇ E. Osaba, E. Onieva, F. Diaz, R. Carballedo, **P. Lopez-Garcia**, A. Perallos, *A migration strategy for distributed evolutionary algorithms based on stopping non-promising subpopulations: A case study on routing problems*, International Journal of Artificial Intelligence.
- ◇ E. Osaba, X.S. Yang, F. Diaz, **P. Lopez-Garcia**, R. Carballedo, *An Improved Discrete Bat Algorithm for Symmetric and Asymmetric Traveling Salesman Problems*, Engineering Applications of Artificial Intelligence. This journal has an impact factor of 2.207 in 2014 Journal Citations Report.

A.2 Conclusions about congestion forecasting

As it has been said along the dissertation, one of the aims to create GACE was to use it to optimize the different parts of the systems that form a hierarchy of FRBSs, i.e. variables, labels, and rules. This optimization was carried out with the objective of predicting congestion at a point or segment in a road. To do that, a total of 12 datasets were created with data of a freeway in California. The prediction horizon takes values of 5, 15, and 30 minutes.

The experimentation was made focusing on the different sizes of the sub-populations in which GA and CE are applied respectively. Different types of datasets were used to prove the performance of the developed algorithm and the hierarchical systems. Two error metrics were used in order to check this performance. With the first one, MAE, GACE with higher size of GA population obtains better results than those with the same sub-population size or a higher CE population size. Also, these results improved the ones obtained by GA and CE. The use of other performance metric was necessary due to MAE does not take into account the differences between the type of predicted congestion and the expected one. As it has been said in previous sections, sMAPE was used to avoid this problem. With the results obtained with this metric, it can be said that the conclusions

A. Conclusions and Future Work

obtained with MAE metrics are correct: GACE with bigger GA population size than CE population size obtains the best errors.

Using sMAPE results, GACE was compared with three literature methods. Looking at the results, it can be said that the proposed technique is a competitive technique. Although, it can improve its performance in several datasets.

Without considering the kind of dataset, the experiment about which configuration of the proposed technique is better for each kind of congestion was made. The conclusions were that higher GA population size is the best option in Free and Severe congestion. These kind of configurations achieved the second place in 3 out of 4 types of congestion. In the case of configurations with a low value of GA population size, these were the best techniques to optimize the systems to forecast Light congestion. In the case of the pure configurations, GA obtained the fourth position as the algorithm to use, while CE was the last option.

Now, taking into account the type of dataset and the predicted congestion value, GACE was the best technique to use in 32 out of 48 cases of study. As it was mentioned before, GACE configurations are the best to predict Null (11 out of 12) and Severe (10 out of 12) congestion, while it is the best technique to forecast Light and Moderate congestions in 6 and 5 out of 12 cases. Despite its bad performance, CE is the chosen algorithm in 8 out of 48, as well as GA.

Finally, about the kind of dataset and prediction horizon, Point Datasets (DP and DPS), and a short-time horizon (5 minutes) obtain the best error values. In the case of Sector Datasets (DS and DSS) it is the opposite: the best values are obtained in the 30-minutes horizon.

A.3 Conclusions about functions optimization

The other part of the experimentation had the aim to prove the performance of the proposed algorithm in different types of optimization functions. With this objective in mind, a total of 24 noise-free functions have been taken from a platform used in different international conferences.

The most important issue in this part of the experimentation was to choose the value for different important parameters as the size of the population, the sizes of the sub-populations, learning rate for the CE part, number of individuals needed to update the means and deviations vectors, and some other parameters. This is important, not only for this problem, but also for future works in different themes. Although the tuning of the parameters depending on the problem is important, with this study, default values for the parameters that assure a good performance in general are found and studied.

First, the size of the population was calculated based on a constant and the dimension of the problem. After that, heat graphics were used to delimitate the values of the size of the GA sub-population, and, besides, the percentage of individuals to update the different parts of CE. With the results obtained, we can assure a greater number of individuals in GA population, and practically the whole CE population used to update means and deviation if the dimension of the problem is low, and the opposite, i.e. small values for both parameters, if the dimension of the problem is high.

As a summary of the results obtained, the proposal improved its results as the dimension of the functions grew. For the lowest dimension used, GACE obtains one of the two best values in 46 % of the functions, while with the highest dimension used, it obtains one of the two best values in 79 % of the functions while the rest of the techniques get worse results, in special the Differential Evolution. For the total 96 cases of study, GACE obtains one of the two best values in 62 cases, while it is the best of all in 29 of them. The second technique which obtains better results is the Differential Evolution, with 37 out of 96, and the best so far in 28 out of 37.

Statistic tests as Friedman, Holm, and Finner were used to assure the results obtained. Using DE as the control algorithm for low dimensions, and GACE for the rest of the dimensions, the result of the tests prove that GACE is significantly different from the rest of the algorithms except one.

A.4 Improvements and future lines of work

Along this chapter, the conclusions extracted from the different experimentations made have been remarked. The algorithm and the FRBSs have obtained a good performance in forecasting congestion, regardless of the type of dataset used, but especially in Point and Section datasets. After that, with the parameters selected in the experimentation studies, the algorithm can reach a value closer to the optimum in optimization functions than several well-known algorithms in the literature. Therefore, we can conclude that the different parts of the hypothesis exposed in Introduction chapter have been accomplished. On the one hand, the proposal solution exposed in this dissertation can achieve a good performance in predicting the different types of congestion that can appear in freeways and help to prevent traffic jams. With this, the first part of the hypothesis is covered. On the other hand, the second part of the hypothesis is concluded with the performance obtained by GACE in optimization function tasks, improving the results reached using well-known literature algorithms as Differential Evolution. Also, values for the different parameters have been established for future problems focusing on their dimension.

But, as it can be seen, the algorithm can still improve its performance in different themes. If we focus on general aspects of the algorithm, improvements to be made can be:

A. Conclusions and Future Work

- ◇ Take into account the parameters as values to optimize along the execution of the algorithm. This issue converts the problem in a multiobjective problem: it is necessary to optimize not only the fitness function, but also the different values of the parameters needed for improving the performance.
- ◇ Diversity in the population. Because it is possible to find the same individuals in different sub-populations, sometimes the diversity of the general population gets lost along the process. This problem can be fixed, for example, increasing the mutation probability along the generations in the GA part, or making a reboot of the individuals in the CE part for updating the mean and deviation vectors.

In the case of the congestion forecasting, the main theme of this dissertation and where the proposal has been applied with more emphasis, due to its importance in real-world problems, the algorithm and the design of the hierarchy can be improved in the following aspects:

- ◇ Improvements in the results obtained for Light and Moderate congestion. As it can be seen in Experimentation chapter, some configurations of the algorithm have problems for making a good forecast in these types of congestion. This could be for the similarities between both values calculation. In future works, the differences between the calculus of the congestion values can be changed in order to improve their correct classification.
- ◇ Use another kind of hierarchy. Serial hierarchy for congestion forecasting has been used in previous works to this dissertation. In our case, the parallel hierarchy has been applied and it has improved the results obtained in other literature works, not only for the use of this kind of hierarchy, but also for the use of the hybrid proposal to optimize its parts. The use of a hybrid hierarchy can be an interesting way to act in order to see the effects of a different way of FRBSs hierarchy.
- ◇ Data from other sources. The different datasets used in this dissertation and the associated paper are available for anyone who wants to use it. There are not many traffic datasets that can be obtained for free. Even so, governmental platforms as the used one (PeMS) allow researches to work with the data they collect. The use of other datasets can certify the good performance of the proposed technique, and it can improve the quality and the different types of traffic datasets.

- ◇ Time horizons. In this dissertation, short-term traffic congestion has been forecasted. The time horizons used have been 5, 15 and 30 minutes. Increasing the number of time horizons, for example, from 5 to 5 between 5 minutes and 60 minutes can specify when the congestion can be forecast with a bigger precision.

In functions optimization, although it is not the principal task of the proposal technique, it has obtained good results in different kind of functions, some of them with a high complexity. Also, it has improved the results obtained by well-known literature algorithms as Differential Evolution. The possible action lines in this theme can be:

- ◇ In the experimentation, several values in which the technique obtains good performance in high dimensions have been used in order to have better results in the dimensions, where other techniques get worse and maintain a certain level of results in low dimensions. The goal now is to adapt these parameters to the dimension, and not have to choose only some of them, as we did with the population size.
- ◇ The use of other types of functions for benchmarking, as well as improving the performance adding or adapting some characteristics of the hybrid method to the functions used, and not only to the operators, can improve the performance and obtain a richer experimentation.

One of the main characteristics of the proposed technique is its adaptation ability to other problems by changing its operators and the codification of the chromosome. Thanks to this, the amount of problems the technique can afford is huge. One of the problems where GACE can be applied is classification using ensembles. Classifier ensembles are sets of individual classifiers that offer a different approach to this problem using clustering for the search space and applying the best individual classifiers to each one of the part of it. In this issue, GACE can be used to improve the different centroids and weights of the areas and the classifiers respectively in order to get a better result with the chosen classifiers of each area.

Another kind of issue, totally different from the exposed in this dissertation, is RFID systems and its tag collision problem. Summarizing, due to the shared nature of the wireless channel used by tags, these systems are prone to transmission collisions, which occurs when two or more tags transmit different information simultaneously. Fuzzy Logic has been applied to this problem in [Arjona 16]. Due to this, it is a good and a different theme where GACE can help in order to find the correct values for different parameters of Fuzzy Logic to get the best performance possible to avoid tag collisions.

A. Conclusions and Future Work

There are always problems to solve. Real world problems such as congestion forecasting, pollution forecasting, and route problems, are current problems in many different projects, as can be found in Horizon 2020 calls, because of their importance in everyday life. Therefore, one of the most important things to do as a researcher is to find possible solutions to these problems. For this purpose, the improvement and development of new algorithms that can deal with a large amount of data, and help to take decisions about important tasks like those mentioned before, are possible ways to present a solution. The proposed method in this dissertation tries to catch these concerns, improving the results given by other techniques. Besides, the development of the proposal demonstrates that not everything is created, and for every problem, there are one or many solutions. Some of them are better than others. Otherwise, other solutions have not been thought yet. This dissertation made a proof of that.

B

Apéndice

B.1 Fórmulas de las funciones utilizadas

Cada tabla muestra las funciones utilizadas agrupadas por su tipo, y su forma matemática de manera que esto pueda ayudar a su entendimiento. En las definiciones de las funciones, se nombra un vector \mathbf{z} que sustituye al argumento x . Este vector es un vector de variable transformada que tiene su óptimo en $\mathbf{z}^{opt} = 0$, si no se indica de otra manera. Para el resto de variables que aparecen en algunas funciones, como son s_i, T_{osz} , ó T_{asy} , su valor se muestra en la documentación que el lector puede encontrar en el siguiente link¹.

Nombre	Fórmula
Sphere	$F_1(x) = x - x^{opt} + F_{opt}$
Ellipsoidal	$F_2(x) = \sum_{i=1}^D 10^6 \frac{i-1}{D-1} z_i^2 + F_{opt}$
Rastrigin	$F_3(x) = 10 \left(D - \sum_{i=1}^D \cos(2\pi z_i) \right) + \ \mathbf{z}\ ^2 + F_{opt}$
Büche-Rastrigin	$F_4(x) = 10 \left(D - \sum_{i=1}^D \cos 2\pi z_i \right) + \sum_{i=1}^D i = 1^D + 100 f_{pen} x + F_{opt}$
Linear Slope	$F_5(x) = \sum_{i=1}^D i = 1^D 5 s_i - s_i z_i + F_{opt}$

Tabla B.1: Funciones Separables: Nombre y fórmulas.

¹ <http://coco.gforge.inria.fr/lib/exe/fetch.php?media=download3.6:bbobdocfunctions.pdf>

B. Apéndice

Nombre	Fórmula
Attractive Sector	$F_6(x) = T_{osz} \left(\sum_{i=1}^D (s_i z_i)^2 \right)^{0.9} + F_{opt}$
Step Ellipsoidal	$F_7(x) = 0.1 \max \left(\frac{ z_1 }{10^4}, \sum_{i=1}^D 10^{2 \frac{i-1}{D-1}} z_i^2 \right) + f_{pen}(x) + F_{opt}$
Rosenbrock , original	$F_8(x) = \sum_{i=1}^{D-1} (100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + F_{opt}$
Rosenbrock , rotated	$F_9(x) = \sum_{i=1}^{D-1} (100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + F_{opt}$

Tabla B.2: Funciones con condicionamiento bajo o moderado: Nombres y funciones.

Nombre	Fórmula
Ellipsoidal	$F_{10}(x) = \sum_{i=1}^D 10^{6 \frac{i-1}{D-1}} z_i^2 + F_{opt}$
Discus	$F_{11}(x) = 10^6 z_1^2 + \sum_{i=2}^D z_i^2 + F_{opt}$
Bent Cigar	$F_{12}(x) = z_1^2 + 10^6 \sum_{i=2}^D z_i^2 + F_{opt}$
Sharp Ridge	$F_{13}(x) = z_1^2 + 100 \sqrt{\sum_{i=2}^D z_i^2} + F_{opt}$
Different Powers	$F_{14}(x) = \sqrt{\sum_{i=1}^D z_i ^{2+4 \frac{i-1}{D-1}}} + F_{opt}$

Tabla B.3: Funciones con condicionamiento elevado y unimodales: Nombres y fórmulas.

Nombre	Fórmula
Rastrigin	$F_{15}(x) = 10 \left(D - \sum_{i=1}^D \cos(2\pi z_i) \right) + \mathbf{z} ^2 + F_{opt}$
Weirstrass	$F_{16}(x) = 10 \left(\frac{1}{D} \sum_{i=1}^D \sum_{k=0}^{11} \frac{1}{2^k} \cos(2\pi 3^k (z_i + \frac{1}{2})) - f_0 \right)^3 + \frac{10}{D} f_{pen}(x) + F_{opt}$
Schaffers F7	$F_{17}(x) = \left(\frac{1}{D-1} \sum_{i=1}^{D-1} \sqrt{s_i} + \sqrt{s_i} \sin^2(50s_i^{1/5}) \right)^2 + 10f_{pen}(x) + F_{opt}$
Schaffers F7, moderately ill-conditioned	$F_{18}(x) = \left(\frac{1}{D-1} \sum_{i=1}^{D-1} \sqrt{s_i} + \sqrt{s_i} \sin^2(50s_i^{1/5}) \right)^2 + 10f_{pen}(x) + F_{opt}$
Composite Griewank-Rosenbrock	$F_{19}(x) = \frac{10}{D-1} \sum_{i=1}^{D-1} \left(\frac{s_i}{4000} - \cos s_i \right) + 10 + F_{opt}$

Tabla B.4: Funciones multimodales con estructura global adecuada: Nombres y fórmulas.

B.1 Fórmulas de las funciones utilizadas

Nombre	Fórmula
Schwefel	$F_{20}(x) = -\frac{1}{D} \sum_{i=1}^D z_i \sin(\sqrt{ z_i }) + 4.189828872724339 + 100f_{pen}(\frac{\mathbf{z}}{100}) + F_{opt}$
Gallagher's Gaussian 101-me Peaks	$F_{21}(x) = T_{osz} \left(10 - \max_{i=1}^{101} w_i \exp \left(-\frac{1}{2D} (x - y_i)^T R^T C_i R (x - y_i) \right) \right)^2 + f_{pen}(x) + F_{opt}$
Gallagher's Gaussian 21-hi Peaks	$F_{22}(x) = T_{osz} \left(10 - \max_{i=1}^{21} w_i \exp \left(-\frac{1}{2D} (x - y_i)^T R^T C_i R (x - y_i) \right) \right)^2 + f_{pen}(x) + F_{opt}$
Katsuura	$F_{23}(x) = \frac{10}{D^2} \prod_{i=1}^D \left(1 + i \sum_{j=1}^{32} \frac{ 2^j z_i - \lfloor 2^j z_i \rfloor }{2^j} \right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} + f_{pen}(x)$
Lunacek bi-Rastrigin	$F_{24}(x) = \min \left(\sum_{i=1}^D (\hat{x}_i - \mu_0)^2, dD + s \sum_{i=1}^D (\hat{x}_i - \mu_1)^2 \right) + 10 \left(D - \sum_{i=1}^D \cos(2\pi z_i) \right) + 10^4 + f_{pen}(x)$

Tabla B.5: Funciones multimodales con estructura global débil: Nombres y fórmulas.



Acrónimos

ACO Optimización de colonia de hormigas, del inglés *Ant Colony Optimization*

ARIMA *Autoregressive Integrated Moving Average*

CE Entropía Cruzada, del inglés *Cross Entropy*

CMA Adaptación de la Matriz de Covarianza, del inglés *Covariance Matrix Adaptation*

DE Evolución Diferencial, del inglés *Differential Evolution*

EDA Algoritmos de Estimación de Distribuciones, del inglés *Estimation of Distribution Algorithms*

FRBS Sistemas basados en Reglas Difusas, del inglés *Fuzzy Rule-Based Systems*

GA Algoritmos Genéticos, del inglés *Genetic Algorithms*

HFRBS Sistemas Basados en Reglas Difusas Jerárquicos, del inglés *Hierarchical Fuzzy Rule-Based Systems*

ITS Sistemas Inteligentes de Transporte, del inglés *Intelligent Transportation Systems*

JSP *Job Scheduling Problem*

KF Filtro de Kalman, del inglés *Kalman Filter*

NN Redes Neuronales, del inglés *Neural Networks*

C. Acrónimos

PSO Algoritmo de Optimización de Partículas, del inglés *Particle Swarm Optimization*

V2V Vehículo a Vehículo

SA Recocido Simulado, del inglés *Simulated Annealing*

SVM Máquinas Vector Soporte, del inglés *Support Vector Machines*

TOD Time-of-Day

TSP *Traveling Salesman Problem*

Bibliografía

- [Abd-El-Wahed 11] W.F. Abd-El-Wahed, A.A. Mousa and M.A. El-Shorbagy. *Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems*. Journal of Computational and Applied Mathematics, Vol. 235, No. 5, pp. 1446–1453, 2011 (pág. 22).
- [Adeli 11] M. Adeli and H. Zarabadipour. *Automatic disease diagnosis systems using pattern recognition based genetic algorithm and neural networks*. International Journal of Physical Sciences, Vol. 6, No. 25, pp. 6076–6081, 2011 (pág. 18).
- [Aja-Fernández 08] S. Aja-Fernández and C. Alberola-López. *Matrix modeling of hierarchical fuzzy systems*. IEEE Transactions on Fuzzy Systems, Vol. 16, No. 3, pp. 585–599, 2008. cited By (since 1996)12 (pág. 29).
- [Alba 05] Enrique Alba. *Parallel metaheuristics: a new class of algorithms*, volume 47. John Wiley & Sons, 2005 (pág. 23).
- [Alba 09] E. Alba and B. Dorronsoro. *Cellular genetic algorithms*, volume 42. Springer-Verlag, 2009 (pág. 85).
- [Alcalá 07] R. Alcalá, J. Alcalá-Fdez and F. Herrera. *A proposal for the genetic lateral tuning of linguistic fuzzy systems and its interaction with rule selection*. IEEE Transactions on Fuzzy Systems, Vol. 15, No. 4, pp. 616–635, 2007 (pág. 58).
- [Arjona 16] L. Arjona, H. Landaluze, A. Perallos and E. Onieva. *Fast fuzzy anti-collision protocol for the RFID standard EPC Gen-2*. IET Electronic Letters, 2016. (Accepted for publication) (pág. 98, 105).

BIBLIOGRAFÍA

- [Awan 11] M.S.K. Awan and M.M. Awais. *Predicting weather events using fuzzy rule based system*. Applied Soft Computing Journal, Vol. 11, No. 1, pp. 56–63, 2011 (pág. 26).
- [Azar 14] A.T. Azar and S.A. El-Said. *Performance analysis of support vector machines classifiers in breast cancer mammography recognition*. Neural Computing and Applications, Vol. 24, No. 5, pp. 1163–1177, 2014 (pág. 35).
- [Bauza 13] R. Bauza and J. Gozalvez. *Traffic congestion detection in large-scale scenarios using vehicle-to-vehicle communications*. Journal of Network and Computer Applications, Vol. 36, No. 5, pp. 1295–1307, 2013 (pág. 3).
- [Benitez 13] A.D. Benitez and J. Casillas. *Multi-objective genetic learning of serial hierarchical fuzzy systems for large-scale problems*. Soft Computing, Vol. 17, No. 1, pp. 165–194, 2013 (pág. 27, 28, 29).
- [Bertsimas 97] D. Bertsimas and J. N. Tsitsiklis. Introduction to linear optimization, volume 6. Athena Scientific, Belmont, MA, 1997 (pág. 5, 32).
- [Bevilacqua 12] Vitoantonio Bevilacqua, Nicola Costantino, Mariagrazia Dotoli, Marco Falagario and Fabio Sciancalepore. *Strategic design and multi-objective optimisation of distribution networks based on genetic algorithms*. International Journal of Computer Integrated Manufacturing, Vol. 25, No. 12, pp. 1139–1150, 2012 (pág. 18).
- [Blum 03] Christian Blum and Andrea Roli. *Metaheuristics in combinatorial optimization: Overview and conceptual comparison*. ACM Computing Surveys (CSUR), Vol. 35, No. 3, pp. 268–308, 2003 (pág. 15).
- [Blum 11] C. Blum, J. Puchinger, G.R. Raidl and A. Roli. *Hybrid metaheuristics in combinatorial optimization: A survey*. Applied Soft Computing Journal, Vol. 11, No. 6, pp. 4135–4151, 2011 (pág. 22).
- [Bouras 13] A. Bouras and W.P. Syam. *Hybrid chaos optimization and affine scaling search algorithm for solving linear programming problems*. Applied Soft Computing Journal, Vol. 13, No. 5, pp. 2703–2710, 2013 (pág. 33).

- [Box 13] George EP Box, Gwilym M Jenkins and Gregory C Reinsel. Time series analysis: forecasting and control. John Wiley & Sons, 2013 (*pág. 3*).
- [Brownlee 11] Jason Brownlee. Clever algorithms: nature-inspired programming recipes. Jason Brownlee, 2011 (*pág. 15*).
- [Calis 15] B. Calis and S. Bulkan. *A research survey: review of AI solution strategies of job shop scheduling problem*. Journal of Intelligent Manufacturing, Vol. 26, No. 5, pp. 961–973, 2015 (*pág. 33*).
- [Casdagli 92] Martin Casdagli. *Chaos and deterministic versus stochastic non-linear modeling*. Journal of the Royal Statistical Society. Series B (Methodological), pp. 303–328, 1992 (*pág. 34*).
- [Chen 04] Y. Chen, J. Dong and B. Yang. *Automatic design of hierarchical TS-FS model using Ant Programming and PSO algorithm*. volume 3192, pp. 285–294, 2004. cited By (since 1996)10 (*pág. 29*).
- [Chen 07] Y. Chen, B. Yang, A. Abraham and L. Peng. *Automatic design of hierarchical Takagi-Sugeno type fuzzy systems using evolutionary algorithms*. IEEE Transactions on Fuzzy Systems, Vol. 15, No. 3, pp. 385–397, 2007. cited By (since 1996)57 (*pág. 29*).
- [Chen 14] Z. Chen and C. Wang. *Research on continuous ant colony optimization algorithm and application in neural network modeling*. Journal of Multiple-Valued Logic and Soft Computing, Vol. 22, No. 3, pp. 317–340, 2014 (*pág. 35*).
- [Coello 02] Carlos A Coello Coello, David A Van Veldhuizen and Gary B Lamont. Evolutionary algorithms for solving multi-objective problems, volume 242. Springer, 2002 (*pág. 34*).
- [Commission 14] European Commission. *Special Eurobarometer 422a, Quality of Transport*, 2014 (*pág. 1*).
- [Cordón 01a] Oscar Cordón. Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases, volume 19. World Scientific, 2001 (*pág. 24, 26*).

BIBLIOGRAFÍA

- [Cordón 01b] Oscar Cordón and Francisco Herrera. *Hybridizing genetic algorithms with sharing scheme and evolution strategies for designing approximate fuzzy rule-based systems*. Fuzzy sets and systems, Vol. 118, No. 2, pp. 235–255, 2001 (pág. 18).
- [Cordón 01c] Oscar Cordón, Francisco Herrera, F Gomide, Frank Hoffmann and Luis Magdalena. *Ten years of genetic fuzzy systems: current framework and new trends*. In IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th, volume 3, pp. 1241–1246. IEEE, 2001 (pág. 18).
- [Cotta-Porras 98] Carlos Cotta-Porras. *Study of hybridization techniques and their application to the design of evolutionary algorithms*. AI Communications, Vol. 11, No. 3, pp. 223–224, 1998 (pág. 23).
- [Crainic 03] Teodor Gabriel Crainic and Michel Toulouse. *Parallel strategies for metaheuristics*. Springer, 2003 (pág. 15).
- [De Boer 05] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor and Reuven Y Rubinstein. *A tutorial on the cross-entropy method*. Annals of operations research, Vol. 134, No. 1, pp. 19–67, 2005 (pág. 20).
- [De Jong 75] Kenneth Alan De Jong. *Analysis of the behavior of a class of genetic adaptive systems*. 1975 (pág. 36, 37).
- [Deb 02] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal and TAMS Meyarivan. *A fast and elitist multiobjective genetic algorithm: NSGA-II*. Evolutionary Computation, IEEE Transactions on, Vol. 6, No. 2, pp. 182–197, 2002 (pág. 35).
- [Deep 10] Kusum Deep and Hadush Mebrahtu Adane. *New Variations of Order Crossover for Travelling Salesman Problem*. International Journal of Combinatorial Optimization Problems and Informatics, Vol. 2, No. 1, pp. 2–13, 2010 (pág. 60).
- [del Jesus 04] M.J. del Jesus, F. Hoffmann, L. Junco and L. Sánchez. *Induction of Fuzzy-Rule-Based Classifiers With Evolutionary Boosting Algorithms*. IEEE Transactions on Fuzzy Systems, Vol. 12, No. 3, pp. 296–308, 2004 (pág. 67).
- [DellÁmico 15] M. DellÁmico, S. Falavigna and M. Iori. *Optimization of a real-world auto-carrier transportation problem*. Transportation Science, Vol. 49, No. 2, pp. 402–419, 2015 (pág. 5).

- [Derrac 11] J. Derrac, S. García, D. Molina and F. Herrera. *A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms*. Swarm and Evolutionary Computation, Vol. 1, No. 1, pp. 3–18, 2011 (pág. 87).
- [Dib 15] O. Dib, M.-A. Manier and A. Caminada. *A Hybrid Metaheuristic for Routing in Road Networks*. volume 2015-October, pp. 765–770, 2015 (pág. 22).
- [Dorigo 97] M. Dorigo and L.M. Gambardella. *Ant colony system: A cooperative learning approach to the traveling salesman problem*. IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1, pp. 53–66, 1997 (pág. 16).
- [Eiben 97] Agoston E Eiben and Thomas Bäck. *Empirical investigation of multiparent recombination operators in evolution strategies*. Evolutionary Computation, Vol. 5, No. 3, pp. 347–365, 1997 (pág. 36, 37).
- [Eiselt 87] H. A. Eiselt, G. Pederzoli and C. Sandblom. Continuous optimization models, volume 1. Walter De Gruyter, 1987 (pág. 5, 32).
- [Elsheikh 14] A. Elsheikh. *Derivative-based hybrid heuristics for continuous-time simulation optimization*. Simulation Modelling Practice and Theory, Vol. 46, pp. 164–175, 2014 (pág. 33).
- [Eshelman 93] Larry J Eshelman. *Real-coded Genetic Algorithms and Interval-Schemata*. Foundations of genetic algorithms, Vol. 2, pp. 187–202, 1993 (pág. 61).
- [Fernández 09] Alberto Fernández, María José del Jesus and Francisco Herrera. *Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets*. International Journal of Approximate Reasoning, Vol. 50, No. 3, pp. 561–577, 2009 (pág. 27).
- [Finner 93] H Finner. *On a monotonicity problem in step-down multiple test procedures*. Journal of the American Statistical Association, Vol. 88, No. 423, pp. 920–923, 1993 (pág. 88).
- [Garcia-Villoria 10] A. Garcia-Villoria, A. Corominas and R. Pastor. *Solving the response time variability problem by means of the cross-entropy method*. International Journal of Manufacturing Technology and Management, Vol. 20, No. 1-4, pp. 316–330, 2010 (pág. 20).

BIBLIOGRAFÍA

- [Genova 15] K. Genova, L. Kirilov and V. Guliashki. *A survey of solving approaches for multiple objective flexible job shop scheduling problems*. Cybernetics and Information Technologies, Vol. 15, No. 2, pp. 3–22, 2015 (pág. 33).
- [Gholamjafari 14] A. Gholamjafari, L. Li, S. Chien and Y. Chen. *Genetic algorithm-based testing scenarios selection for the performance evaluation of crash imminent braking systems for pedestrian safety*. pp. 1656–1661, 2014 (pág. 18).
- [Ghorbani 10] Reza Ghorbani, Eric Bibeau and Shaahin Filizadeh. *On conversion of hybrid electric vehicles to plug-in*. IEEE Transactions on Vehicular Technology, Vol. 59, No. 4, pp. 2016–2020, 2010 (pág. 26).
- [Glover 86] Fred Glover and Claude McMillan. *The general employee scheduling problem. An integration of MS and AI*. Computers & operations research, Vol. 13, No. 5, pp. 563–573, 1986 (pág. 16).
- [Glover 06] Fred W Glover and Gary A Kochenberger. Handbook of metaheuristics, volume 57. Springer Science & Business Media, 2006 (pág. 14, 15).
- [Goldberg 91] David E Goldberg and Kalyanmoy Deb. *A comparative analysis of selection schemes used in genetic algorithms*. Urbana, Vol. 51, pp. 61801–2996, 1991 (pág. 60).
- [Haber 10] R.E. Haber, R.M. Del Toro and A. Gajate. *Optimal fuzzy control system using the cross-entropy method. A case study of a drilling process*. Information Sciences, Vol. 180, No. 14, pp. 2777–2792, 2010 (pág. 20).
- [Hansen 01] N. Hansen and A. Ostermeier. *Completely derandomized self-adaptation in evolution strategies*. Evolutionary Computation, Vol. 9, No. 2, pp. 159–195, 2001 (pág. 4, 16).
- [Hansen 09] Nikolaus Hansen, Steffen Finck, Raymond Ros and Anne Auger. *Real-parameter black-box optimization benchmarking 2009: Noisy functions definitions*. 2009 (pág. 36).
- [He 13] K. He, Y. Jin and W. Huang. *Heuristics for two-dimensional strip packing problem with 90 rotations*. Expert Systems with Applications, Vol. 40, No. 14, pp. 5542–5550, 2013 (pág. 33).

- [Hernández 13] S.A. Hernández, G. Leguizamón and E. Mezura-Montes. *Hybridization of Differential Evolution using Hill Climbing to solve Constrained Optimization Problems*. Revista Iberoamericana de Inteligencia Artificial, Vol. 16, No. 52, pp. 3–15, 2013 (pág. 21, 22).
- [Herrera 98] Francisco Herrera, Manuel Lozano and Jose L. Verdegay. *Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis*. Artificial intelligence review, Vol. 12, No. 4, pp. 265–319, 1998 (pág. 61).
- [Holland 75] J. H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. Univ. of Michigan Press, 1975 (pág. 16, 18, 33).
- [Holm 79] S. Holm. *A simple sequentially rejective multiple test procedure*. Scandinavian Journal of Statistics, pp. 65–70, 1979 (pág. 88).
- [Holtschulte 13] N. J. Holtschulte and M. Moses. *Benchmarking cellular genetic algorithms on the BBOB noiseless testbed*. In Proceedings of the Companion Publication of the 2013 Conference on Genetic and Evolutionary Computation, pp. 1201–1208. ACM, 2013 (pág. 85).
- [Hyndman 06] R.J. Hyndman and A.B. Koehler. *Another look at measures of forecast accuracy*. International Journal of Forecasting, Vol. 22, No. 4, pp. 679–688, 2006 (pág. 65).
- [Ianovsky 11] E. Ianovsky and J. Kreimer. *An optimal routing policy for unmanned aerial vehicles (analytical and cross-entropy simulation approach)*. Annals of Operations Research, Vol. 189, No. 1, pp. 215–253, 2011 (pág. 20).
- [Iglesias 10] José Antonio Iglesias, Plamen Angelov, Agapito Ledezma and Araceli Sanchis. *Human activity recognition based on evolving fuzzy systems*. International Journal of Neural Systems, Vol. 20, No. 05, pp. 355–364, 2010 (pág. 26).
- [Jacobson 04] S. H. et al. Jacobson. *Global optimization performance measures for generalized hill climbing algorithms*. Journal of Global Optimization, Vol. 29, No. 2, pp. 173–190, 2004 (pág. 85).

BIBLIOGRAFÍA

- [Jelleli 10] T. Jelleli and A. Alimi. *Automatic design of a least complicated hierarchical fuzzy system*. In: 6th IEEE World Congress on computational intelligence, pp. 1–7, 2010. cited By (since 1996)1 (pág. 29).
- [Jia 06] Lei Jia, Licai Yang, Qingjie Kong and Shu Lin. *Study of artificial immune clustering algorithm and its applications to urban traffic control*. International Journal of Information Technology, Vol. 12, No. 3, pp. 1–6, 2006 (pág. 3).
- [Jiang 08] Chao Jiang, X Han, GR Liu and GP Liu. *A nonlinear interval number programming method for uncertain optimization problems*. European Journal of Operational Research, Vol. 188, No. 1, pp. 1–13, 2008 (pág. 33).
- [Joo 09] M.G. Joo and T. Sudkamp. *A method of converting a fuzzy system to a two-layered hierarchical fuzzy system and its run-time efficiency*. IEEE Transactions on Fuzzy Systems, Vol. 17, No. 1, pp. 93–103, 2009. cited By (since 1996)10 (pág. 29).
- [Jourdan 09] L. Jourdan, M. Basseur and E.-G. Talbi. *Hybridizing exact methods and metaheuristics: A taxonomy*. European Journal of Operational Research, Vol. 199, No. 3, pp. 620–629, 2009 (pág. 23, 33).
- [Kanarachos 15] S. Kanarachos and A. Kanarachos. *Intelligent road adaptive suspension system design using an experts'based hybrid genetic algorithm*. Expert Systems with Applications, Vol. 42, No. 21, pp. 8232–8242, 2015 (pág. 18).
- [Keller 85] James M Keller, Michael R Gray and James A Givens. *A fuzzy k-nearest neighbor algorithm*. Systems, Man and Cybernetics, IEEE Transactions on, No. 4, pp. 580–585, 1985 (pág. 56).
- [Kennedy 10] J. Kennedy. *Particle swarm optimization*. In Encyclopedia of Machine Learning, pp. 760–766. Springer-Verlag, 2010 (pág. 16).
- [Klir 95] George Klir and Bo Yuan. *Fuzzy sets and fuzzy logic, volume 4*. Prentice hall New Jersey, 1995 (pág. 24).
- [Kullback 51] S. Kullback and R. A. Leibler. *On information and sufficiency*. The annals of mathematical statistics, Vol. 22, No. 1, pp. 79–86, 1951 (pág. 19).

- [Kuo 15] R.J. Kuo, Y.H. Lee, F.E. Zulvia and F.C. Tien. *Solving bi-level linear programming problem through hybrid of immune genetic algorithm and particle swarm optimization algorithm*. Applied Mathematics and Computation, Vol. 266, pp. 1013–1026, 2015 (pág. 35).
- [Lagarias 98] J.C. Lagarias, J.A. Reeds, M.H. Wright and P.E. Wright. *Convergence properties of the Nelder-Mead simplex method in low dimensions*. SIAM Journal on Optimization, Vol. 9, No. 1, pp. 112–147, 1998 (pág. 33).
- [Larranaga 99] Pedro Larranaga, Cindy M. H. Kuijpers, Roberto H. Murga, Inaki Inza and Sejla Dizdarevic. *Genetic algorithms for the travelling salesman problem: A review of representations and operators*. Artificial Intelligence Review, Vol. 13, No. 2, pp. 129–170, 1999 (pág. 61).
- [Lebacque 09] JP Lebacque, TY Ma and MM Khoshyaran. *The cross-entropy field for multimodal dynamic assignment*. Proceedings of the Traffic and Granular Flow, 2009 (pág. 4).
- [Lema 95] Ninatubu Mbora Lema and Andrew DF Price. *Benchmarking: performance improvement toward competitive advantage*. Journal of Management in Engineering, Vol. 11, No. 1, pp. 28–37, 1995 (pág. 36).
- [Li 07] Q. Li, G. Liu, W. Zhang, C. Zhao, Y. Yin and Z. Wang. *A specific genetic algorithm for optimum path planning in intelligent transportation system*. pp. 140–143, 2007 (pág. 18).
- [Liao 14] T. Liao, T. Stützle, M.A. Montes De Oca and M. Dorigo. *A unified ant colony optimization algorithm for continuous optimization*. European Journal of Operational Research, Vol. 234, No. 3, pp. 597–609, 2014 (pág. 35).
- [Liu 15] S.-Y. Liu, D.-W. Li, Y.-G. Xi and Q.-F. Tang. *A short-term traffic flow forecasting method and its applications*. Journal of Shanghai Jiaotong University (Science), Vol. 20, No. 2, pp. 156–163, 2015 (pág. 22).
- [Lopez-Garcia 15] Pedro Lopez-Garcia, Enrique Onieva, Eneko Osaba, Antonio D. Masegosa and Asier Perallos. *Hybridizing Genetic Algorithm with Cross Entropy for Solving Continuous Functions*. In Proceedings of the Companion Publication

BIBLIOGRAFÍA

- of the 2015 Conference on Genetic and Evolutionary Computation, GECCO Companion '15, pp. 763–764, 2015 (pág. 47).
- [Lopez-Garcia 16a] P. Lopez-Garcia, E. Onieva, E. Osaba, A. D. Masegosa and A. Perallos. *A Hybrid Method for Short-Term Traffic Congestion Forecasting Using Genetic Algorithms and Cross Entropy*. IEEE Transactions on Intelligent Transportation Systems, Vol. 17, No. 2, pp. 557–569, 2016 (pág. 29, 46).
- [Lopez-Garcia 16b] P. Lopez-Garcia, E. Onieva, E. Osaba, A.D. Masegosa and A. Perallos. *GACE: A meta-heuristic based in the hybridization of Genetic Algorithms and Cross Entropy methods for continuous optimization*. Expert Systems with Applications, Vol. 55, pp. 508–519, 2016 (pág. 47).
- [López 13] Victoria López, Alberto Fernández, Salvador García, Vasile Palade and Francisco Herrera. *An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics*. Information Sciences, Vol. 250, pp. 113–141, 2013 (pág. 55).
- [Lozano 11] Manuel Lozano, Daniel Molina and Francisco Herrera. *Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems*. Soft Computing, Vol. 15, No. 11, pp. 2085–2087, 2011 (pág. 36, 37).
- [Lu 14] Y. Lu, Y. Liu, C. Gao, L. Tao and Z. Zhang. *A novel Physarum-Based ant colony system for solving the real-world traveling salesman problem*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 8794, pp. 173–180, 2014 (pág. 5).
- [Luaces 03] O. Luaces. *Inflating examples to obtain rules*. International Journal of Intelligent Systems, Vol. 18, pp. 1113–1143, 2003 (pág. 68).
- [Mandal 11] A. Mandal, A.K. Das, P. Mukherjee, S. Das and P.N. Suganthan. *Modified differential evolution with local search algorithm for real world optimization*. pp. 1565–1572, 2011 (pág. 22).

- [Mansoori 08] Eghbal G Mansoori, Mansoor J Zolghadri and Seraj D Katebi. *SGERD: A steady-state genetic algorithm for extracting fuzzy classification rules from data*. IEEE Transactions on Fuzzy Systems, Vol. 16, No. 4, pp. 1061–1071, 2008 (pág. 67).
- [Muñoz 13] Mario A Muñoz, Michael Kirley and Saman K Halgamuge. *The algorithm selection problem on the continuous optimization domain*. In Computational Intelligence in Intelligent Data Analysis, pp. 75–89. Springer, 2013 (pág. 40).
- [Mühlenbein 93] Heinz Mühlenbein and Dirk Schlierkamp-Voosen. *Predictive models for the breeder genetic algorithm i. continuous parameter optimization*. Evolutionary computation, Vol. 1, No. 1, pp. 25–49, 1993 (pág. 61).
- [Murat 14] Y.S. Murat, S. Kutluhan and N. Uludag. *Use of fuzzy optimization and linear goal programming approaches in urban bus lines organization*. Advances in Intelligent Systems and Computing, Vol. 223, pp. 377–387, 2014 (pág. 33).
- [Neri 10] F. Neri and V. Tirronen. *Recent advances in differential evolution: A survey and experimental analysis*. Artificial Intelligence Review, Vol. 33, No. 1-2, pp. 61–106, 2010 (pág. 16).
- [Ning 15] Z. Ning, C. Tao, L. Fei and X. Haitao. *A Hybrid Heuristic Algorithm for the Intelligent Transportation Scheduling Problem of the BRT System*. Journal of Intelligent Systems, Vol. 24, No. 4, pp. 437–448, 2015 (pág. 22).
- [Nocedal 06] J. Nocedal and S. J. Wright. Numerical optimization. Springer, 2006 (pág. 33).
- [Okulewicz 13] M. Okulewicz and J. Mańdziuk. *Application of particle swarm optimization algorithm to dynamic vehicle routing problem*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 7895 LNAI, No. PART 2, pp. 547–558, 2013 (pág. 35).
- [Onieva 09] E. Onieva, J. Alonso, J. Perez, V. Milanés and T. de Pedro. *Autonomous car fuzzy control modeled by iterative genetic algorithms*. In FUZZ-IEEE 2009. IEEE International Conference on, pp. 1615–1620, Aug 2009 (pág. 26).

BIBLIOGRAFÍA

- [Onieva 12] Enrique Onieva, Vicente Milanés, Jorge Villagra, Joshué Pérez and Jorge Godoy. *Genetic optimization of a vehicle fuzzy decision system for intersections*. Expert Systems with Applications, Vol. 39, No. 18, pp. 13148–13157, 2012 (pág. 18).
- [Osaba 13] Eneko Osaba, Enrique Onieva, Roberto Carballedo, Fernando Diaz, Asier Perallos and Xiao Zhang. *A multi-crossover and adaptive island based population algorithm for solving routing problems*. Journal of Zhejiang University SCIENCE C, Vol. 14, No. 11, pp. 815–821, 2013 (pág. 18, 19).
- [Osaba 14] Eneko Osaba, Fernando Diaz and Enrique Onieva. *Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts*. Applied Intelligence, pp. 1–22, 2014 (pág. 18, 33, 35).
- [Papadimitriou 98] C. H. Papadimitriou and K. Steiglitz. Combinatorial optimization: Algorithms and complexity. Courier Corporation, 1998 (pág. 5, 32).
- [Posawang 10] P. Posawang, S. Phosaard, W. Polnigongit and W. Pattara-Atikom. *Perception-based road traffic congestion classification using neural networks and decision tree*. Lecture Notes in Electrical Engineering, Vol. 60, pp. 237–248, 2010 (pág. 4).
- [Pošík 12] Petr Pošík and Václav Klemš. *Benchmarking the differential evolution with adaptive encoding on noiseless functions*. In Proceedings of the 14th annual conference companion on Genetic and evolutionary computation, pp. 189–196. ACM, 2012 (pág. 84).
- [Purwar 15] A. Purwar and S.K. Singh. *Hybrid prediction model with missing value imputation for medical data*. Expert Systems with Applications, Vol. 42, No. 13, pp. 5621–5631, 2015 (pág. 21).
- [Qiao 11] J. Qiao, N. Yang and J. Gao. *Two-stage fuzzy logic controller for signalized intersection*. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, Vol. 41, No. 1, pp. 178–184, 2011 (pág. 18).
- [Quinlan 86] J. Ross Quinlan. *Induction of decision trees*. Machine learning, Vol. 1, No. 1, pp. 81–106, 1986 (pág. 67).
- [Quinlan 14] J Ross Quinlan. C4. 5: programs for machine learning. Elsevier, 2014 (pág. 67).

- [Raidl 06] G.R. Raidl. *A unified view on hybrid metaheuristics*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 4030 LNCS, pp. 1–12, 2006 (pág. 23).
- [Rubinstein 97] R.Y. Rubinstein. *Optimization of computer simulation models with rare events*. European Journal of Operational Research, Vol. 99, No. 1, pp. 89–112, 1997 (pág. 19).
- [Rubinstein 99] R. Rubinstein. *The cross-entropy method for combinatorial and continuous optimization*. Methodology and Computing in Applied Probability, Vol. 1, No. 2, pp. 127–190, 1999 (pág. 4, 16, 20).
- [Sánchez 01] L. Sánchez, I. Couso and J.A. Corrales. *Combining GP Operators With SA Search To Evolve Fuzzy Rule Based Classifiers*. Information Sciences, Vol. 136, No. 1-4, pp. 175–192, 2001 (pág. 67).
- [Sangsawang 14] C. Sangsawang, K. Sethanan, T. Fujimoto and M. Gen. *Metaheuristics optimization approaches for two-stage reentrant flexible flow shop with blocking constraint*. Expert Systems with Applications, Vol. 42, No. 5, pp. 2395–2410, 2014 (pág. 21).
- [Sawyer 13] B. A. Sawyer, A. O. Adewumi and M.M. Ali. *Benchmarking projection-based real coded genetic algorithm on BBOB-2013 noiseless function testbed*. In Proceedings of the Companion Publication of the 2013 Conference on Genetic and Evolutionary Computation, pp. 1193–1200, 2013 (pág. 84).
- [Schwefel 81] H. Schwefel. Numerical optimization of computer models. John Wiley & Sons, 1981 (pág. 5, 32).
- [Segura 15] Carlos Segura, Carlos A Coello Coello and Alfredo G Hernández-Díaz. *Improving the vector generation strategy of Differential Evolution for large-scale optimization*. Information Sciences, Vol. 323, pp. 106–129, 2015 (pág. 22).
- [Shimajima 95] K. Shimajima, T. Fukuda and Y. Hasegawa. *Self-tuning fuzzy modeling with adaptive membership function, rules, and hierarchical structure based on genetic algorithm*. Fuzzy Sets and Systems, Vol. 71, No. 3, pp. 295–309, 1995. cited By (since 1996)101 (pág. 29).

BIBLIOGRAFÍA

- [Silva 14] E. Silva, J.F. Oliveira and G. Wäscher. *2DCPackGen: A problem generator for two-dimensional rectangular cutting and packing problems*. European Journal of Operational Research, Vol. 237, No. 3, pp. 846–856, 2014 (pág. 33).
- [Skycomp 09] Cox Skycomp Inc. in association with Whytney Bailey and LLC Magnani. *Major Highway Performance Ratings and Bottleneck Inventory, State of Maryland - Spring 2008*. 2009 (pág. 52).
- [Sousa 04] Tiago Sousa, Arlindo Silva and Ana Neves. *Particle swarm based data mining algorithms for classification tasks*. Parallel Computing, Vol. 30, No. 5, pp. 767–783, 2004 (pág. 67).
- [Stellet 15] J.E. Stellet, F. Straub, J. Schumacher, W. Branz and J.M. Zollner. *Estimating the Process Noise Variance for Vehicle Motion Models*. pp. 1512–1519, 2015 (pág. 4).
- [Syberfeldt 09] A. Syberfeldt, A. Ng, R.I. John and P. Moore. *Multi-objective evolutionary simulation-optimisation of a real-world manufacturing problem*. Robotics and Computer-Integrated Manufacturing, Vol. 25, No. 6, pp. 926–931, 2009 (pág. 5).
- [Talbi 02] E-G Talbi. *A taxonomy of hybrid metaheuristics*. Journal of heuristics, Vol. 8, No. 5, pp. 541–564, 2002 (pág. 4, 22, 23, 39, 42).
- [Talbi 09] El-Ghazali Talbi. *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons, 2009 (pág. 14).
- [Tan 07] M.-C. Tan, L.-B. Feng and J.-M. Xu. *Traffic flow prediction based on hybrid ARIMA and ANN model*. Zhongguo Gonglu Xuebao/China Journal of Highway and Transport, Vol. 20, No. 4, pp. 118–121, 2007 (pág. 3).
- [Toledo 14] C.F.M. Toledo, L. Oliveira and P.M. França. *Global optimization using a genetic algorithm with hierarchically structured population*. Journal of Computational and Applied Mathematics, Vol. 261, pp. 341–351, 2014 (pág. 35).
- [Topcuoglu 07] H. R. Topcuoglu, B. Demiroz and M. Kandemir. *Solving the register allocation problem for embedded systems using a hybrid evolutionary algorithm*. IEEE Transactions on Evolutionary Computation, Vol. 11, No. 5, pp. 620–634, 2007 (pág. 21).

- [Valizadeh 14] M. Valizadeh, S. Syafie and I.S. Ahamad. *Multi-objective particle swarm optimization for optimal planning of biodiesel supply chain in Malaysia*. Advances in Soft Computing, Vol. 287, pp. 293–302, 2014 (pág. 35).
- [Van Laarhoven 87] P. JM Van Laarhoven and E. HL Aarts. Simulated annealing. Springer-Verlag, 1987 (pág. 16).
- [Verdegay 08] José L Verdegay, Ronald R Yager and Piero P Bonissone. *On heuristics as a fundamental constituent of soft computing*. Fuzzy Sets and Systems, Vol. 159, No. 7, pp. 846–855, 2008 (pág. 13).
- [Wang 04] ZG Wang, YS Wong and M Rahman. *Optimisation of multi-pass milling using genetic algorithm and genetic simulated annealing*. The International Journal of Advanced Manufacturing Technology, Vol. 24, No. 9-10, pp. 727–732, 2004 (pág. 22, 40).
- [Wang 05] Yibing Wang and Markos Papageorgiou. *Real-time freeway traffic state estimation based on extended Kalman filter: a general approach*. Transportation Research Part B: Methodological, Vol. 39, No. 2, pp. 141–167, 2005 (pág. 3).
- [Wang 09] Y. Wang and Z. Cai. *A hybrid multi-swarm particle swarm optimization to solve constrained optimization problems*. Frontiers of Computer Science in China, Vol. 3, No. 1, pp. 38–52, 2009 (pág. 5).
- [Wang 13] J. Wang and Q. Shi. *Short-term traffic speed forecasting hybrid model based on Chaos-Wavelet Analysis-Support Vector Machine theory*. Transportation Research Part C: Emerging Technologies, Vol. 27, pp. 219–232, 2013 (pág. 4).
- [Welch 95] Greg Welch and Gary Bishop. *An introduction to the Kalman filter*, 1995 (pág. 3).
- [Whitley 95] D. Whitley, K. Mathias, S. Rana and J. Dzuber. *Building Better Test Functions*. In Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 239–246. Morgan Kaufmann, 1995 (pág. 36).
- [Wolpert 97] David H Wolpert and William G Macready. *No free lunch theorems for optimization*. Evolutionary Computation, IEEE Transactions on, Vol. 1, No. 1, pp. 67–82, 1997 (pág. 36).

BIBLIOGRAFÍA

- [Wright 99] S. J. Wright. *Continuous Optimization (Nonlinear and Linear Programming)*. Foundations of Computer-Aided Process Design, 1999 (pág. 33).
- [Wright 05] M.H. Wright. *The interior-point revolution in optimization: History, recent developments, and lasting consequences*. Bulletin of the American Mathematical Society, Vol. 42, No. 1, pp. 39–56, 2005 (pág. 33).
- [Xia 12] M. Xia and Z. Xu. *Entropy/cross entropy-based group decision making under intuitionistic fuzzy environment*. Information Fusion, Vol. 13, No. 1, pp. 31–47, 2012 (pág. 20).
- [Zadeh 65] Lotfi A Zadeh. *Fuzzy sets*. Information and control, Vol. 8, No. 3, pp. 338–353, 1965 (pág. 24).
- [Zadeh 94] Lotfi A. Zadeh. *Soft computing and fuzzy logic*. IEEE Software, Vol. 11, No. 6, pp. 48–56, 1994 (pág. 13).
- [Zadeh 96] Lotfi A Zadeh. *Fuzzy logic= computing with words*. Fuzzy Systems, IEEE Transactions on, Vol. 4, No. 2, pp. 103–111, 1996 (pág. 24).
- [Zargari 12] Shahriar Afandizadeh Zargari, Salar Zabihi Siabil, Amir Hossein Alavi and Amir Hossein Gandomi. *A computational intelligence-based approach for short-term traffic flow prediction*. Expert Systems, Vol. 29, No. 2, pp. 124–142, 2012 (pág. 4).
- [Zhang 09] T. Zhang, Y.-T. Li, R. Yan, C.-G. Zheng and Z. Yang. *Multiscale cross entropy and its application in analyzing the nonlinear coupling pattern between sympathetic flow and blood pressure*. Chinese Journal of Biomedical Engineering, Vol. 28, No. 5, pp. 652–657, 2009 (pág. 20).
- [Zhang 12a] J. Zhang and R. Lv. *Fuzzy particle swarm optimization algorithm in solving traveling salesman problem*. International Review on Computers and Software, Vol. 7, No. 5, pp. 2593–2597, 2012 (pág. 33).
- [Zhang 12b] L. Zhang, J. Ma and C. Zhu. *Theory modeling and application of an adaptive Kalman filter for short-term traffic flow prediction*. Journal of Information and Computational Science, Vol. 9, No. 16, pp. 5101–5109, 2012 (pág. 2, 4).

- [Zhang 14] Xiao Zhang, Enrique Onieva, Asier Perallos, Eneko Osaba and Victor Lee. *Hierarchical fuzzy rule-based system optimized with genetic algorithms for short term traffic congestion prediction*. Transportation Research Part C: Emerging Technologies, 2014 (pág. 4, 29).
- [Zheng 14] L.-J. Zheng, D.-C. Dong and D.-Y. Wang. *A hybrid intelligent algorithm for the vehicle scheduling problems with time windows*. pp. 2756–2761, 2014 (pág. 22).
- [Zou 13] G. Zou and R. Kulkarni. *A systematic genetic algorithm based framework to optimize Intelligent Transportation System (ITS) strategies*. 2013 (pág. 18).