UNIVERSIDAD
DE MÁLAGA

TESIS DOCTORAL

Diseño, Implementación y Evaluación de Nuevos Algoritmos Evolutivos para Problemas Dinámicos

Design, Implementation, and Evaluation of New Evolutionary Algorithms for Dynamic Problems

Tesis en Ingeniería del Software e Inteligencia Artificial

Autor: Mg. Yesnier Bravo García

Directores: Dr. Enrique Alba Torres y Dr. Gabriel J. Luque Polo

E.T.S.I. Informática
Departamento de Lenguajes y Ciencias de la Computación

Universidad de Málaga (España)

Enero 2017

UNIVERSIDAD DE MÁLAGA

AUTOR: Yesnier Bravo García

http://orcid.org/0000-0002-6030-1484

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

Departamento de Lenguajes y Ciencias de la Computación

Escuela Técnica Superior de Ingeniería Informática

Universidad de Málaga

El Dr. Enrique Alba Torres y el Dr. Gabriel J. Luque Polo, pertenecientes al Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga,

Certifican

que, D. Yesnier Bravo García, Magíster en Informática por la Universidad de Málaga, ha realizado en el Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, bajo su dirección, el trabajo de investigación correspondiente a su Tesis Doctoral (por compendio de artículos) titulada:

Diseño, Implementación y Evaluación de Nuevos Algoritmos Evolutivos para Problemas Dinámicos

Revisado el presente trabajo, estimamos que puede ser presentado al tribunal que ha de juzgarlo. Y para que conste a efectos de lo establecido en la legislación vigente, autorizamos la presentación de esta Tesis Doctoral en la Universidad de Málaga.

En Málaga, enero de 2017

Fdo: Dr. Enrique Alba Torres y Dr. Gabriel J. Luque Polo

A mi *descendencia*, Olivia.

AGRADECIMIENTOS

Esta tesis doctoral no hubiese llegado a su fin sin la invaluable ayuda de muchas personas que me acompañaron en el proceso.

En primer lugar, me gustaría agradecer a mis directores de tesis, los profesores Enrique Alba y Gabriel Luque, por su dedicación y paciencia para transitar conmigo este camino lleno de obstáculos. A la familia NEO, a los que hoy estamos y a los que hemos pasado por aquí: Jamal, Daniel, Cristian, Javier Ferrer, Javier Arellano, Andrés, Francis, David, Hajer, Martín, Zakaria, Kamel, Paco, Pablo, Sergio, Raúl, Sofiane, entre otros. Todos han sido parte importante de mi vida, gracias por acogerme y por aguantarme tanto tiempo.

Agradezco también a los amigos con los que he tenido el placer de coincidir y convivir en Málaga, fuera del contexto universitario: Ana Rosario, Fabián, Luis, Yordanis, Ana Sanchez, Mariela, Fernanda, Arturo, Moatez, Valerio, Keamo, Mpo, Mehdi, y a muchos otros; a ustedes agradezco mi mente sana y mi salud no tanto.

A mi familia de Marbella Elena, Pepe, José, Isabel, y al pequeño Alejandro; por su cercanía, por el abrigo que me brindaron.

Quiero agradecer profundamente a mi familia, por su indispensable apoyo desde la distancia. Especialmente a mi Madre, que ha sido mi sostén en todo momento. A mi Padre, a mis hermanos Arelis Ketty, Claudia y Sergio, a mi prima Leyvis, por las infinitas charlas.

No pueden faltar los agradecimientos a mi esposa Ailema, por su amor, por su infinita espera, y por su comprensión.

Y sobre todo a mi hija Olivia, desde que llegaste a este mundo yo soy infinitamente más fuerte.

ÍNDICE

1 INTRODUCCIÓN

Día a día estamos expuestos a complejos problemas de optimización, en muchos casos con incertidumbre y dinamismo. En el ámbito de las Tecnologías Informáticas, un problema de optimización consiste en encontrar la "mejor" configuración de un conjunto de variables o parámetros de decisión en un problema lo más real posible (Goldberg 1989; Glover and Kochenberger 2003). La mejor solución (u óptimo) para un problema de optimización global (mono-objetivo) es aquella que maximiza (o minimiza) una función de evaluación (función objetivo) sujeta a un conjunto de restricciones (lineales o no). En el dominio de los algoritmos de optimización, a esta función se la denomina a veces función de *fitness*, ya que asigna a cada solución candidata un valor de calidad que luego se usa para guiar al método (normalmente iterativo) que busca el óptimo. En muchos casos, el espacio de búsqueda (o espacio de soluciones) es tan complejo que no es suficiente aplicar técnicas tradicionales basadas en enumeración sin una guía inteligente como motor de la búsqueda. Los problemas reales difícilmente presentan funciones de evaluación simples, sino que con frecuencia incluyen grandes dimensiones, ausencias de algunos datos, complejas restricciones e incluso pueden estar sujetos a incertidumbre.

De hecho, tradicionalmente la literatura considera que un problema de optimización mantiene su definición y sus datos durante todo el tiempo en que un algoritmo lo está resolviendo. Estos problemas estáticos son los más usuales y han recibido gran atención en el pasado (Talbi 2009; Alba et al. 2009). Sin embargo, en la actualidad también son muy comunes y de gran interés, problemas que cambian algunas de sus características a medida que el algoritmo que los soluciona progresa (Jin and Branke 2005; Nguyen, Yang, and Branke 2012; Rohlfshagen, Lehre, and Yao 2013; Alba, Nakib, and Siarry 2013). En estos problemas, la función que se está optimizando depende del tiempo, por lo que cuando se consulta la calidad de una misma solución tentativa facilitada por el algoritmo, el resultado de la evaluación puede variar dependiendo del momento en que se realice.

En general, los comportamientos dinámicos ocurren cuando las condiciones o datos que teníamos deben incorporar algún tipo de cambio que, además, es imprescindible abordar para la resolución final del problema. Ejemplos de esto son cuando aparecen datos o prioridades de último minuto durante la resolución del problema, o cuando se da un retraso en la recepción de materias primas, o se produce la rotura eventual de máquinas industriales, la demanda cambiante de los clientes, entre otros muchos casos. En realidad, la búsqueda de un óptimo

se complica de esta manera, al tiempo que traslada el típico enfoque en conseguir calidad y buenas prestaciones al final del proceso en un nuevo dominio de trabajo, donde importa mucho (y se usa como resultado parcial) lo que ocurre durante el proceso de optimización. De hecho, es posible que no exista un final determinado para tal proceso, sino que haya que estar constantemente buscando óptimos una y otra vez por tiempo indeterminado. Por ello, surgen nuevas inquietudes, métricas, formas de abordar estos problemas y es necesario diseñar, implementar y evaluar nuevos algoritmos: el objetivo principal de esta tesis.

Todas estas consideraciones en la definición de un problema pueden verse como un tipo de *incertidumbre*. Los investigadores distinguen distintos tipos de problemas con incertidumbre, incluyendo ruido, imprecisión en los datos o mediciones y dinamismo (Jin 2004; Jin and Branke 2005). Estas incertidumbres han llevado a definir distintos tipos de problemas de optimización. En problemas con ruido, evaluar la misma solución dos veces puede devolver valores diferentes (por ejemplo, debido a mediciones de equipos industriales o simulaciones). La imprecisión en los datos o mediciones normalmente derivan en problemas de aproximación o robustez. Éstos surgen por la dificultad de evaluar de la función o el escenario real. En estos casos, se suele emplear una función aproximada en lugar de la función real (*surrogates*), usualmente basada en simulaciones o datos experimentales, y encontrar soluciones que sigan siendo buenas a pequeñas variaciones del problema (Jin 2005). Con lo cual, el escenario real con incertidumbre se reduce a un problema de optimización *estático*, es decir, que no cambia de cara al método o algoritmo utilizado para resolverlo.

En contraste, los problemas de optimización dinámicos (*Dynamic Optimization Problems* o DOP), se caracterizan por tener en cuenta los cambios del problema a lo largo de la optimización (Jin and Branke 2005; Nguyen, Yang, and Branke 2012). En dichos problemas, el objetivo, a diferencia de los problemas estáticos, ya no es encontrar el óptimo en cada instante, sino "seguir" (*track*) los movimientos de dicho óptimo en el espacio de búsqueda asociado, lo más cerca posible a través del tiempo. Este tipo de problemas tiene mayor presencia en aplicaciones reales y ofrece mayor grado de dificultad que los antes mencionados. De hecho, los DOP no tienen una única solución (como en un problema de aproximación o robusto) sino una serie de soluciones a lo largo de múltiples periodos estacionarios (dentro de los cuales el problema no cambia).

El objeto de esta investigación doctoral se centra en los DOP. El método de solución más simple de un DOP parece ser reiniciar la búsqueda cada vez que ocurre un cambio en el problema. Sin embargo, la alta complejidad y dimensionalidad de los DOP cuando nos interesan escenarios reales hace impracticable reiniciar la búsqueda cada vez que ocurre un cambio en el problema. La razón es que este reinicio impide reutilizar la información aprendida sobre el problema previamente. En la comunidad científica se acepta que cada nuevo escenario

guarda cierta relación con el anterior (Jin and Branke 2005; Cruz, González, and Pelta 2011; Nguyen, Yang, and Branke 2012). En caso contrario (cada escenario es totalmente independiente de todo lo anterior), incluso pudiera ser más apropiado usar un método aleatorio (Branke and Schmeck 2003), por la frecuencia con que se presentan los cambios del problema. Por lo tanto, el algoritmo debe adaptarse continuamente a una solución óptima que cambia a lo largo del proceso de optimización, lo cual incorpora nuevas dificultades. Por ejemplo, zonas del espacio de búsqueda actualmente prometedoras pueden dejar de serlo tras el cambio, o los cambios pueden ocurrir con alta frecuencia, necesitando una rápida adaptación al nuevo entorno, entre otros.

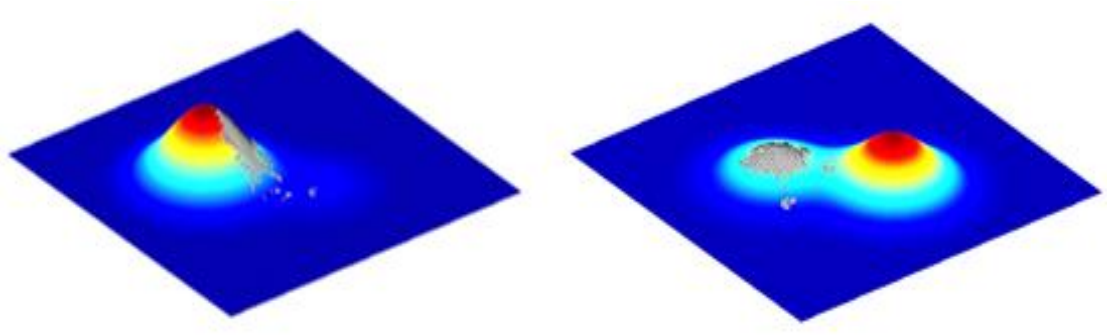
Por otro lado, como ya se comentó, la resolución de problemas complejos del mundo real (sean DOP o no), requiere de técnicas especiales para obtener soluciones precisas en un tiempo razonable. Para resolver problemas reales, un acercamiento muy popular es crear técnicas inspiradas en procesos de la Naturaleza, tales como la *selección natural* o el comportamiento colectivo de los pájaros. El conjunto de técnicas creadas conforman el campo de la Computación Evolutiva (EC, Evolutionary Computation en inglés) (Bäck, Fogel, and Michalewicz 1997), que incluye los Algoritmos Evolutivos (EA), y otras metaheurísticas tales como Optimización basada en Enjambres de Partículas (PSO, Particle Swarm Optimization en inglés) (Kennedy and Eberhart 1995) o métodos basados en Colonias de Hormigas (ACO, Ant Colony Optimization) (Dorigo, Maniezzo, and Colorni 1996).

Actualmente, las técnicas de EC se aplican en muchos dominios, tales como el aprendizaje máquina, robótica, logística, procesamiento de señales digitales, juegos, diseño óptimo de vehículos, sistemas de control y planificación, por mencionar algunos ejemplos (Alba et al. 2009). La razón es que estos algoritmos ofrecen muchas ventajas con respecto a los convencionales (Fogel 1997; Yao and Xu 2006):

- Simplicidad conceptual y computacional
- Requisitos mínimos de información acerca del problema que resolver
- Robustez
- Paralelismo natural
- Amplia aplicabilidad
- Rendimiento adecuado en problemas de dimensiones reales
- Permite incorporar conocimiento del dominio e integrar con otros métodos
- Capacidad de adaptación
- No requiere expresiones analíticas ni derivaciones matemáticas

Dichas ventajas hacen que el campo de la EC sea también muy apropiado para resolver DOP. Sin embargo, en la práctica, las técnicas de EC también presentan serias dificultades para adaptarse a los cambios continuos del problema

(Branke 2001; Morrison 2004; De Jong 1999). En general, las técnicas de EC son métodos iterativos basados en poblaciones de soluciones candidatas, que emplean algún tipo de operador de variación y aceptación para guiar la búsqueda de mejores soluciones para un problema. Como resultado, la población tiende a *converger*, es decir, a estabilizarse próximo a la mejor solución encontrada hasta el momento. Sin embargo, una vez que la población ha convergido, es difícil reaccionar y adaptarse de forma apropiada a los cambios en el problema (movimiento del óptimo en el espacio de búsqueda). La Figura 1 ilustra el problema de la convergencia.



a) Antes del cambio ***b)*** Tras el cambio

Figura 1. Ilustración del problema de la convergencia en EDO. La población (puntos grises en la figura) converge de forma natural al óptimo actual (a), lo cual le impide reaccionar y adaptarse al problema tras el cambio (b).

Un gran número de trabajos de investigación se han dedicado a buscar mecanismos para solventar esta limitación, dando lugar al campo de la *Optimización Dinámica Evolutiva* (Evolutionary Dynamic Optimization o EDO), o el campo de estudio y aplicación de las técnicas de EC para DOP (Branke 2001; Jin 2004). Sin embargo**,** a pesar de contar con importantes avances o contribuciones, el campo de la EDO aún se considera muy reciente y presenta importantes desafíos (Nguyen, Yang, and Branke 2012), entre los que se pueden destacar los siguientes:

- **Estudios metodológicos.** En la literatura existe un gran número de métodos para resolver el problema de la convergencia de los EA en DOP, así como formas diferentes de implementarlos (Cruz, González, and Pelta 2011). Sin embargo, la falta de estudios integrales sobre los métodos existentes hace que se desconozca la verdadera contribución de cada implementación al rendimiento del algoritmo: faltan *metodologías unificadas* para el uso de los distintos métodos. Además, muchos de estos métodos se han propuesto para tratar características específicas de ciertos DOP, tales como la existencia de múltiples soluciones óptimas o la recurrencia de escenarios similares. Sin embargo, en la práctica, los DOP pueden

presentar distintos comportamientos dinámicos a la vez, incluyendo algunos nuevos nunca antes vistos. Por lo tanto, se necesitan más estudios que examinen los parámetros y el rendimiento de los métodos actuales ante diferentes tipos de DOP.

- **Estudios teóricos.** Los estudios de teoría en EDO son muy escasos cuando se compara con otros campos relacionados con la optimización (Rohlfshagen, Lehre, and Yao 2013). Realizar estudios teóricos es muy importante para entender tanto el comportamiento de los DOP como el rendimiento de los métodos propuestos. Por ejemplo, si los investigadores tuvieran una métrica precisa para reconocer alguna característica dinámica en un DOP, podrían seleccionar el método más apropiado para resolverlo. Por otro lado, los fundamentos teóricos no sólo ayudarían a explicar los estudios experimentales actuales, sino que los mejorarían. Por ejemplo, se podrían reducir considerablemente los tiempos de investigación y utilizar eficientemente los recursos computacionales. De hecho, los análisis experimentales a menudo no son suficiente para sacar conclusiones; se necesitan conocimientos teóricos (teoremas, definiciones, modelos, etc.) para entender los resultados obtenidos.

- **Estudios de validación y aplicación a problemas reales**. La separación entre los estudios actuales, en su mayoría realizados en base a problemas académicos, y las aplicaciones a problemas reales de la industria es otra de las preocupaciones en EDO. Los trabajos actuales suelen hacer asunciones sobre los DOP que solo se presentan en la práctica en algunos casos (Cheng and Yang 2010; Li-Ning Xing et al. 2011; Cheng, Yang, and Wang 2012). En este sentido, primero se precisa definir las características comunes de las plataformas de prueba, así como los criterios que se usan para la evaluación de los métodos actuales. Luego, es preciso estudiar cómo estas características y criterios se reflejan en escenarios de aplicación reales.

Esta tesis doctoral explora las posibilidades de contribuir al campo de EDO en las tres áreas antes mencionadas. En primer lugar, estudiamos dos de los métodos propuestos en EDO para resolver el problema de la convergencia: los algoritmos paralelos distribuidos (*Parallel Distributed EA* o dEA) y los métodos basados en *memoria global*. Por un lado, los dEA estructuran la población en *islas*, que evolucionan de forma independiente y se comunican mediante *migraciones* de individuos (Homayounfar, Areibi, and Wang 2003; Cheng and Yang 2010) . Por otro lado, los métodos basados en memoria global almacenan explícitamente las buenas soluciones encontradas (posibles óptimos después de un cambio) y las utilizan para guiar la búsqueda (Branke 1999).

Actualmente, existen varios algoritmos en EDO que emplean múltiples poblaciones (islas en un dEA) e intercambian algún tipo de información (realizan migraciones) (Oppacher and Wineberg 1999; Branke et al. 2000; Ursem 2000; Park, Choe, and Ryu 2008). Asimismo, un número importante de algoritmos en EDO integran esquemas de memoria global (Brest et al. 2009; Halder, Das, and Maity 2013; Lepagnot et al. 2013; K. Trojanowski and Michalewicz 1999; Yang 2008; Yang and Li 2010). Sin embargo, el rendimiento de cada método depende de la configuración de algunos parámetros o decisiones de diseño claves, como son las *políticas de migración* o los criterios de *reemplazo* y *recuperación* de soluciones de la memoria. Hasta donde tenemos conocimiento, no hay un estudio integral sobre los beneficios y limitaciones de cada variante, para resolver distintos tipos de DOP.

En relación a la falta de estudios teóricos, exploramos la posibilidad de analizar la convergencia en EDO a partir del *takeover time* (Goldberg and Deb 1991), una métrica ampliamente usada en problemas estáticos, que requieren una definición clara para DOP. El *takeover time* (tiempo en que la mejor solución absorbe al resto de la población) mide la presión de selección (en qué medida se favorecen las mejores soluciones) en un EA (Miller and Goldberg 1995). Esto es importante en EDO, dado que una convergencia muy rápida o muy lenta hace que el algoritmo no pueda encontrar y perseguir la mejor solución. Hasta donde conocemos, nunca nadie había definido takeover time para DOP, donde la "mejor solución" cambia constantemente.

Por último, también en esta tesis abordamos el tercer gran desafío en EDO, relacionado con la aplicación de técnicas de EDO a problemas reales de nuestras ciudades. Las investigaciones aquí presentadas están más relacionadas con el trabajo futuro, que en gran medida ya se ha iniciado para seguir desarrollando las líneas de impacto claras de aplicación real que se derivan de esta tesis. Por tal razón, en algunos casos no se trata el problema directamente como dinámico, sino que nos enfocamos en entender el problema y desarrollar algoritmos bases de optimización evolutivas que después serán mejorados con las técnicas aquí desarrolladas. En otros, se hace una interpretación de algunos fundamentos desarrollados también como parte de nuestra investigación y que tienen una aplicación directa en escenarios reales, tales como la noción de *takeover time*.

1.1 Objetivos y metodología de la investigación

Una vez presentado el objeto de estudio de esta tesis y las áreas donde se enmarcan los principales desafíos (los detalles siguen en las siguientes secciones), abordamos ahora directamente la definición de los objetivos de la tesis y la metodología adoptada como guía para el logro de los mismos.

El *objetivo* general de esta tesis doctoral es diseñar, implementar y evaluar nuevos EA para problemas de optimización dinámicos. Las necesidades actuales en investigación, el interés de esta combinación de teoría y práctica, y los beneficios que se adivinan gracias a este trabajo justifican el interés de este objetivo. No sólo se trata de un objetivo de base científica, sino que pretendemos obtener resultados con impacto en la práctica diaria de este dominio, así como ampliar sus horizontes de aplicación al mundo real en la medida de lo posible.

Para alcanzar el objetivo general antes mencionado lo subdividimos en los siguientes objetivos específicos:

1. Analizar el rendimiento de los dEA (nuevos algoritmos), con diferentes configuraciones de parámetros en la *política de migración*, ante DOP con diferentes características.
2. Evaluar el rendimiento de los métodos de *memoria* global (nuevas técnicas), con diferentes estrategias de *reemplazo* y *recuperación* de soluciones, en problemas dinámicos con diferentes características.
3. Establecer un modelo de convergencia en EDO (nuevos fundamentos) basado en una definición clara de takeover time y la construcción de modelos matemáticos para calcular su valor teniendo en cuenta las características de los DOP.
4. Crear nuevas técnicas en EDO (prestaciones mejoradas para enfrentar problemas reales), usando resultados experimentales o teóricos, para aplicarlas a problemas reales de optimización en Movilidad Inteligente.

Para guiar nuestro camino hasta estos objetivos, en el sentido que marca el *Método Científico* (Popper 1999), planteamos una serie de preguntas de investigación *(Research Questions* o RQ) que puedan luego validar y confirmar nuestra hipótesis de trabajo en forma de una tesis final: *los nuevos algoritmos diseñados tienen base teórica aplicada a la práctica y eso los hace más eficientes que el estado del arte*. En concreto se plantean seis preguntas de investigación distribuidas en los tres ámbitos mencionados (véase Figura 2).

Los estudios realizados en esta investigación se centran fundamentalmente en EA, que son en general algoritmos más adaptables para la resolución de problemas reales de distintos tipos, en especial, problemas con comportamiento dinámico. Ello se debe, entre otros factores, a una mayor flexibilidad en la resolución y la robustez de una búsqueda global. Por otro lado, enfocamos el estudio a problemas que tienen un solo objetivo, aunque muchos resultados pueden también aplicarse a problemas multiobjetivos.

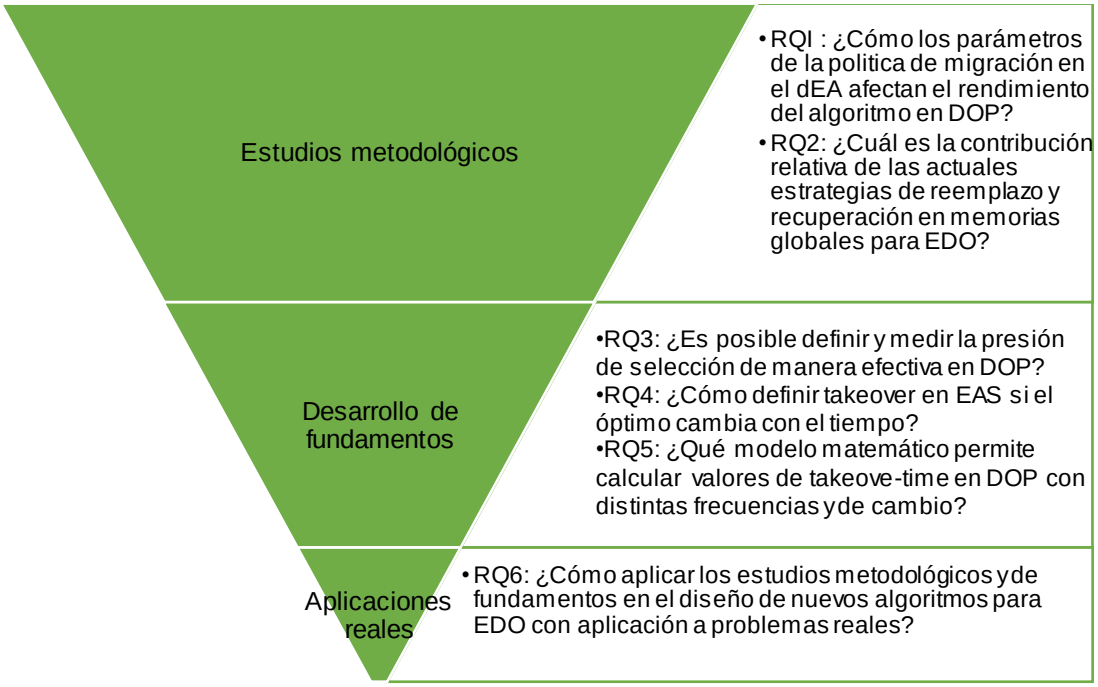


Figura 2. Preguntas de investigación que nos guían hasta el objetivo final.

1.2 Publicaciones y contribuciones

Para terminar esta introducción, haremos un resumen de las publicaciones y contribuciones a las que nos ha llevado la metodología propuesta, todas ellas directamente relacionadas con las áreas de desafíos, el objetivo marcado y guiados por las preguntas de investigación anteriores.

Durante el desarrollo de esta tesis se han realizado un total de seis publicaciones, que a modo de referencia y muy tempranamente en esta memoria, se listan a continuación:

1. Y. Bravo, G. Luque, E. Alba. *Influence of the Migration Period in Parallel Distributed GAs for Dynamic Optimization*. Learning and Intelligent Optimization (LION), 6th International Conference, pp. 343–348, Springer Berlin Heidelberg, 2012. ISSN 0302-9743
2. Y. Bravo, G. Luque, E. Alba. *Migrants Selection and Replacement in Distributed Evolutionary Algorithms for Dynamic Optimization*. Distributed Computing and Artificial Intelligence (DCAI). LNCS, pp. 155–162, Springer, 2013. ISSN 0302-9743
3. Y. Bravo, G. Luque, E. Alba. *Global memory schemes for dynamic optimization*. Natural Computing 15 (2): 319–33. ISSN 1567-7818
4. Y. Bravo, G. Luque, E. Alba. *Takeover time in dynamic optimization problems*. Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), pp. 25–30, IEEE, 2013. ISBN: 978-1-4673-5849-1

5. Y. Bravo, G. Luque, E. Alba. *Takeover Time in Evolutionary Dynamic Optimization: From theory to practice*. Applied Mathematics and Computation (AMC) 250: 94–104, 2015. ISSN 0096-3003.
6. Y. Bravo, J. Ferrer, G. Luque, E. Alba. *Smart Mobility by Optimizing the Traffic Lights: A New Tool for Traffic Control Centers*. Smart-CT 2016, LNCS 9704, pp. 147–156, Springer Berlin Heidelberg, 2016, ISSN: 0302-9743

Las contribuciones *originales* de esta tesis, a lo largo de las publicaciones antes mencionadas, son:

1. Estudio sobre la influencia de las políticas de migración en el rendimiento de los dEA en DOP, identificando correlaciones existentes entre los parámetros que controlan el rendimiento del algoritmo y las características los DOP (Véase la Publicación 1 y 2).
2. La propuesta de un modelo unificado para los métodos basados en memoria global en EDO, que considera las estrategias de reemplazo y recuperación como parámetro (Véase la Publicación 3).
3. La definición de métricas de rendimiento para evaluar los métodos basados en memoria global para EDO (Véase la Publicación 3).
4. Análisis de la influencia de las principales cuestiones de diseño de los esquemas de memoria global, señalando las ventajas y desventajas relativas a cada estrategia para hacer frente a distintas características del problema dinámico (Véase la Publicación 3).
5. Definición de takeover time para problemas dinámicos (donde la "mejor solución" cambia continuamente) que caracteriza la convergencia de los algoritmos evolutivos en estos problemas (Véase la Publicación 4).
6. El desarrollo de modelos matemáticos para calcular valores precisos de takeover time, siguiendo la definición propuesta y a partir de características del algoritmo y características del problema (Véanse las Publicaciones 4 y 5).
7. Aplicación de técnicas de computación evolutiva y resultados de estudios metodológicos y de fundamentos a problemas reales en el dominio de Movilidad Inteligente (Véase la Publicación 6).

Las publicaciones realizadas avalan las contribuciones de esta tesis y su impacto en la comunidad científica. Dos de estas publicaciones (números 3 y 5) se han incluido en revistas internacionales bien posicionadas (categorías Q1 y Q2) en la base de datos del Instituto para la Información Científica (Institute for Scientific Information o ISI). Mientras que el resto de las publicaciones se han presentado en congresos internacionales de gran interés en las áreas abordadas (véase Anexo A).

1.3 Estructura de la memoria

De acuerdo con la normativa vigente, esta Tesis doctoral se presenta como *compendio de publicaciones*. Por lo que, además de proponer para su evaluación un conjunto de trabajos publicados que cumplen con los requisitos establecidos, se presenta esta memoria que está estructurada como sigue.

Tras este apartado introductorio para describir la motivación de nuestro trabajo, los objetivos y la metodología utilizada, y para presentar las publicaciones y contribuciones realizadas, pasamos a una parte técnica que aborda los preliminares científicos de esta investigación donde se justifican los objetivos de investigación. Concretamente, el Capítulo 2 muestra detalles sobre el estado actual de cada una de las tres áreas de desafíos claves en EDO, e incluye una síntesis de los conceptos necesarios para entender los estudios realizados y los resultados obtenidos.

El Capítulo 3 presenta un resumen global de los resultados obtenidos. El propósito de este capítulo es dar una explicación especial conjunta de las publicaciones que avalan esta tesis para adaptar este texto al formato de *compendio de artículos*, formato elegido para defender esta tesis doctoral.

Finalmente, en el Capítulo 4 se revisan y discuten los resultados obtenidos y sus implicaciones, tratando de dar una visión integradora. Al mismo tiempo, se enuncian una serie de trabajos futuros para dar continuidad a la línea de investigación desarrollada.

2 PRELIMINARES CIENTÍFICOS DE ESTA TESIS DOCTORAL

El uso de Algoritmos Evolutivos (*Evolutionary Algorithms* o EA) para resolver DOP, u Optimización Dinámica Evolutiva (*Evolutionary Dynamic Optimization* o EDO), se ha incrementado notablemente en las últimas dos décadas. Como se ha señalado anteriormente, estas técnicas tienen características favorables para resolver DOP (simplicidad, robustez, paralelismo natural, aplicabilidad, etc.), pero también algunas limitaciones. Hasta la fecha, se han logrado importantes avances en la definición y caracterización de DOP, en los métodos para hacer frente a las limitaciones de las técnicas convencionales, en las plataformas de prueba y métricas de rendimiento para el estudio y comparación de estos nuevos enfoques; también destacan algunas contribuciones teóricas. Sin embargo, el campo tiene aún muchas líneas de investigación abiertas, de acuerdo con el criterio de la mayoría de los investigadores implicados (Jin and Branke 2005; Cruz, González, and Pelta 2011; Nguyen, Yang, and Branke 2012) que se mencionaron en el primer apartado de este documento.

En esta sección se describen algunos fundamentos y avances en el campo de la EDO que sirven de base para entender los resultados de esta investigación. En concreto, en el primer epígrafe se incluyen algunas definiciones básicas sobre EDO, mientras que en los restantes se aportan los fundamentos necesarios para posteriormente contestar a las preguntas de investigación indicadas en el capítulo introductorio de esta tesis.

2.1 Optimización dinámica evolutiva: Metodologías

Pasemos a realizar algunas definiciones útiles básicas para entender nuestro trabajo. Concretamente, vamos a introducir formalmente los problemas dinámicos, enumerando los criterios que se usan comúnmente para caracterizarlos, así como los tipos de técnicas propuestas para mejorar el desempeño de los Algoritmos evolutivos en DOP.

Para ello, primero es necesario conocer qué es un problema de optimización global, definido formalmente como sigue.

Definición 1 (Optimización Global): Un *problema de optimización* se define como:

$$\max_{x \in S} \{ f(x), sujeto\ a\ g_j(x) \geq 0 \}, \forall\ j = 1,2,3\ldots. \quad (1)$$

donde S es el espacio de búsqueda, $f: S \rightarrow \Re$ es la función objetivo, g_j para $j = 1,2,3,\ldots$ es un conjunto de restricciones que debe cumplir x. Una solución óptima, u *óptimo*, es $x^* \in S$ tal que $f(x) \leq f(x^*)$ para todo $x \in \aleph(x^*)$, donde $\aleph \subseteq S$ representa una vecindad de x^* en S. Si $\aleph = S$ entonces x^* es un óptimo *global*, de lo contrario es un óptimo *local*.

■

La Definición 1 describe un problema de maximización, pero esto no resta generalización ya que un problema de minimización puede expresarse también de esta forma: $min\ f(x) \equiv max[-f(x)]$. Por otro lado, aquí se considera un solo objetivo, también existen problemas que optimizan varios objetivos a la vez (multi-objetivos), donde $f(x) = [f_1(x), f_2(x), \ldots, f_k(x)]^T$ es una familia de k funciones objetivo, pero cuyo estudio está fuera del alcance de esta tesis. Así mismo, distintos tipos de problemas de optimización surgen a partir de variar el dominio del espacio de búsqueda, por ejemplo, combinatorio ($S = Z^n$) o contínuo ($S = \Re^n$), siendo n el número de variables de decisión.

Sobre todo, la definición anterior hace referencia a un problema de optimización *estático*, donde la función objetivo, las variables de decisión y las restricciones no cambian durante la ejecución del algoritmo. Si alguno de estos elementos cambia a menudo, entonces podría ser un problema de optimización dinámico, en lo adelante *problema dinámico*, que es el objeto de estudio de la presente tesis doctoral.

2.1.1 Problemas Dinámicos

La literatura existente suele definir problema de optimización dinámico (DOP) de varias maneras: **(1)** como un problema de optimización que tiene parámetros dependientes del tiempo en su expresión matemática (Bosman 2005; Woldesenbet and Yen 2009), o **(2)** como una secuencia de problemas de optimización estática conectados por algunas reglas dinámicas (Weicker 2000; Rohlfshagen, Lehre, and Yao 2013; Rohlfshagen and Yao 2008). Basados en estas propuestas, otros autores han propuesto nuevas formulaciones que consideran características específicas de los problemas como, por ejemplo, la *dependencia temporal* entre los distintos escenarios del problema a lo largo plazo (Nguyen and Yao 2009). Recientemente, se ha hecho un esfuerzo por lograr una definición unificada, basada en la idea de toma de decisiones múltiples (Haobo et al. 2014).

En general, en la comunidad se acepta comúnmente que los algoritmos deben resolver un DOP *online*, es decir, dar una solución para cada estado del problema sin información de los cambios futuros. Además, se menciona explícitamente la relación entre dos estados consecutivos del problema (Branke and Schmeck 2003; Nguyen, Yang, and Branke 2012). Por lo tanto, el algoritmo puede "aprender" del pasado y utilizar esa información en el seguimiento del óptimo después de cada cambio. Teniendo en cuenta lo antes planteado se introduce la siguiente definición:

Definición 2 (Optimización Dinámica): Un *problema de optimización dinámico*, *variable en el tiempo* o *no estacionario* se define como:

$$\max_{x \in S} \{ f_t(x), sujeto\ a\ {g_t}^j(x) \geq 0 \}, \forall i, j = 1,2,3\ldots;\ t \in T_i \qquad (2)$$

donde $f_t : S \rightarrow \Re$ es una función objetivo estática para todo $t \in T_i$ ($i = 1,2,3,\ldots$), siendo T_i el *intervalo estacionario*; cuyas soluciones deben satisfacer las restricciones ${g_j}^t (j = 1,2,3,\ldots)$. El objetivo aquí es encontrar una secuencia de *óptimos* ${x^*}_1, {x^*}_2, {x^*}_3, \ldots$ en S tal que $f_t(x) \leq f_t({x^*}_i)$ para todo $t \in T_i$.
■

La complejidad global de abordar un DOP es mayor que para un problema de optimización global estático (de hecho, engloba normalmente varios problemas dinámicos que resolver). Esto se debe a que, en lugar de encontrar una única solución a lo largo de la ejecución, requiere encontrar una secuencia de soluciones. Esta secuencia de óptimos es representativa de los cambios continuos del problema, de modo que, además de enfrentarse a las dificultades topológicas del espacio de búsqueda, el algoritmo también tiene la dificultad de la *dinámica de cambio*. El siguiente apartado examina algunas de estas dinámicas de cambio usadas en la literatura para caracterizar DOP.

2.1.2 Características de los problemas dinámicos

De acuerdo con De Jong (1999), los problemas de optimización dinámicos no cambian aleatoriamente, sino que exhiben ciertos patrones de comportamiento que los caracterizan. Varios autores han propuesto taxonomías o métricas diferentes para caracterizar un problema dinámico. Weicker (2000) propuso primero la descomposición de un DOP en componentes que cambian siguiendo **(1)** una secuencia de transformaciones de coordenadas (por ejemplo, traslación lineal o rotación alrededor de una centro) y **(2)** re-escalado del *fitness*. Además, consideró seis clases de DOP dependiendo de posibles combinaciones de estos dos criterios. Yaniasaki et al. (2002) da otra distinción de DOP basada en características típicas de los DOP: reaparición, continuidad, rareza y previsibilidad. Pero la

caracterización más extendida en la actualidad sigue la propuesta inicial de Branke (1999), a partir del diseño de un generador de problemas dinámicos:

- **Periodo de cambio**: Indica cada cuántas unidades de tiempo ocurren los cambios. Los periodos largos/cortos representan cambios lentos/rápidos, respectivamente.
- **Severidad de cambio**: Describe cuán diferente es el espacio de soluciones después de un cambio. Pequeños cambios resultan en movimientos continuos y graduales del óptimo, mientras que valores altos de este parámetro suele relacionarse con cambios drásticos del problema.
- **Periodicidad**: Caracteriza los entornos *recurrentes* en los que un número finito de estados se repiten durante el proceso de optimización. Esta característica tiene dos rasgos distintivos: *longitud del ciclo*, que es el tiempo que tarda el DOP para volver a un estado anterior; y *precisión del ciclo*, o grado de similitud con el estado anterior.
- **Predictibilidad**: Si los cambios en el problema son predecibles o no. Los cambios *caóticos* son más difíciles predecir.
- **Dependencia temporal**: Indica si la solución actual afecta o no a los cambios futuros del problema.

De acuerdo con varios autores en el campo (De Jong 1999; Nguyen, Yang, and Branke 2012; Yang, Jiang, and Nguyen 2013), no está claro si estos criterios son suficientes para caracterizar todos los DOP que existen. Sin embargo, estos criterios abarcan tipos distintos de comportamientos dinámicos y permiten distinguir grados de dificultad diferentes para los distintos métodos de optimización dinámica que existen.

2.1.3 Mejoras en los EA para la resolución de DOP

Para resolver DOP con EA los investigadores han ideado distintos métodos para enfrentar el problema de la convergencia (véase Figura 3). El primer método es generar diversidad en la población después de cada cambio (Cobb 1990). El segundo mecanismo consiste en mantener la diversidad a lo largo de toda la ejecución del algoritmo (Grefenstette 1992). La tercera técnica consiste en el uso de memoria (Branke 1999) para guiar la búsqueda después de un cambio a partir de información del pasado. El cuarto método usa múltiples poblaciones para buscar y perseguir muchos óptimos candidatos a la vez (Park and Ryu 2007; Ursem 2000; Oppacher and Wineberg 1999; Branke et al. 2000). Otro acercamiento propuesto persigue el aprendizaje y la predicción de los cambios (Bosman 2007) también en base a la experiencia con escenarios anteriores del problema. Finalmente, algunos métodos híbridos combinan una o más métodos anteriores (Ayvaz, Topcuoglu, and Gurgen 2006; Brest et al. 2009; Halder, Das, and Maity 2013; Lepagnot et al. 2013; Krzysztof Trojanowski and Michalewicz 1999; Li and Yang 2008b; Yang and Li 2010; Topcuoglu, Ucar, and Altin 2014)

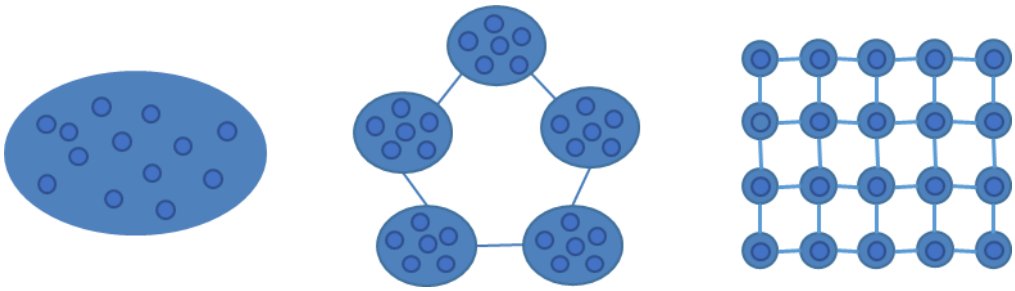
Figura 3. Taxonomía de métodos usados en EDO.

Cada método propuesto para EDO tiene sus ventajas y sus limitaciones (Nguyen et al. 2013). Además, desde el punto de vista metodológico, algunos métodos (el uso de múltiples poblaciones y el uso de memoria) requieren mayor estudio en DOP. Por un lado, los métodos multi-poblacionales no han sido estudiado lo suficiente, en concreto, el algoritmo evolutivo paralelo distribuido. En segundo lugar, se han propuesto muchas variantes de memoria, específicamente, tipos de memoria global, sin que exista un análisis sobre la contribución relativa de cada diseño. En ambos casos destacan las capacidades para diseñar nuevos algoritmos capaces de adaptarse a distintas dinámicas de cambio. Las siguientes secciones describen estos dos métodos, respectivamente.

2.1.3.1 Algoritmos evolutivos distribuidos para EDO

Los EA paralelos distribuidos (dEA) han demostrado ser muy efectivos en escenarios de altos requerimientos de diversidad y recursos de cómputo (Luque and Alba 2011); sin embargo, su uso en EDO está por debajo de lo esperado (RQ1). Actualmente, muchos trabajos consideran aplicar métodos híbridos y auto-adaptativos, que detectan cambios en el problema y reaccionan ajustando sus parámetros de evolución. Muchos de ellos usan múltiples poblaciones para explorar distintas soluciones sub-óptimas al problema (posibles óptimos después de un cambio) al mismo tiempo. En este sentido, uno de los dominios que aún requieren estudio en EDO son los *algoritmos evolutivos paralelos estructurados* (PEA, por sus siglas en inglés).

Los EA no pueden adaptarse bien a los cambios del problema por su modelo de población en *panmixia*. En este modelo cualesquiera dos soluciones, llamadas *individuos*, tienen iguales probabilidades de cruzarse y sobrevivir (enfatizando el proceso de selección natural). Para resolver esta limitación, los PEA estructuran la población siguiendo alguna distribución espacial de los individuos (véase Figura 4).



a) Panmixia ***b)*** Distribuido ***c)*** Celular

Figura 4. Principales modelos de estructuración de la población en EAs.

En particular, los algoritmos paralelos distribuidos (dEA), descentralizan la población en subpoblaciones o *deme*, llamadas *islas* (ver Figura 4 b). Cada isla evoluciona de forma independiente, posiblemente en paralelo, y se comunica con las islas vecinas mediante *migraciones* de individuos. Las migraciones de individuos entre dos islas consisten en la selección de ciertos individuos de la primera para su incorporación en otra isla vecina.

El rendimiento de un dEA depende del ajuste adecuado de algunos parámetros que se describen a continuación:

- **Topología**: Específica las conexiones o vecinos de cada isla (es decir, los orígenes y destinatarios de las migraciones). Algunas topologías populares en la literatura son: Estrella, Anillo y completamente conectado.
- **Homogeneidad**: Sí todas las islas evolucionan de la misma forma, es decir, con la misma configuración de parámetros, el dEA se denomina *homogéneo*, de lo contrario este se denomina *heterogéneo*.
- **Periodo de migración (ζ)**: es el número de generaciones que ocurren entre dos migraciones consecutivas de una isla.
- **Tasa de migración (m)**: Representa el número de individuos enviados desde una isla otra, a menudo definido como un porcentaje de del tamaño de la población.
- **Criterios de selección (ω_S)** y **reemplazo (ω_R)**: Estos criterios determinan cómo se seleccionan las soluciones que emigran desde la isla origen y cómo éstas son incorporadas en la isla destino, respectivamente.

- **Sincronización (**$sync|async$**)**: Indica si las islas **(1)** tras enviar, deben esperar por las soluciones que recibidas de sus vecinos (síncrono) o **(2)** el proceso envío y recepción están desacoplados y no se debe esperar a los individuos de otras islas, sino que las va insertando en la población cada vez conforme las reciba (asíncrono).

Varios autores han analizado previamente la influencia de las políticas de migración en problemas estáticos. Por ejemplo, en Cantú-Paz (1999) y Alba y Troya (2000) demostraron la conveniencia de migrar un individuo al azar en lugar de la mejor solución de cada isla.

Los métodos en EDO basados en múltiples poblaciones también emplean algún intercambio de información (realizan migraciones usando políticas o estrategias muy concretas). Por ejemplo, Oppacher y Wineberg (1999) envían los individuos *élite* (mejores) de las subpoblaciones llamadas *colonias* a una subpoblación central. Otras políticas utilizadas en la literatura implican un conocimiento de toda la población, como en el método *multinacional* propuesto por Ursem en (2000), aplicando el mecanismo de detección de valles (óptimos locales) entre los mejores individuos de cada subpoblación, denominada *nación*. Más recientemente, Park, Choe y Ryu (2008) proponen usar dos poblaciones con estrategias de evolución complementarias (explotación vs diversificación) y, dada la inconveniencia de las migraciones normales, aplican el cruzamiento entre soluciones de distintas poblaciones como forma de intercambio de información.

Sin embargo, hasta donde tenemos conocimiento, muy pocos autores han propuesto usar dEA (Homayounfar, Areibi, and Wang 2003; Cheng and Yang 2010). Tampoco existe un estudio sobre la influencia que tienen las políticas de migración que emplean los actuales métodos multi-poblacionales en EDO: ¿Con qué frecuencia debería intercambiarse información entre las islas? ¿Cómo deberíamos seleccionar las soluciones que migran o se aceptan en cada isla? ¿El mismo criterio aplica para distintos tipos de DOP?

Las respuestas a estas cuestiones (RQ1) se trataron en los artículos (1) y (2) del compendio que avala la presente tesis doctoral (véase Sección 3.1 y Sección 3.2, respectivamente). Primero en (1) se analiza la influencia del *periodo de migración* (cada cuánto tiempo se intercambia información entre las islas), uno de los parámetros claves en la política de migración. En la segunda publicación sobre parámetros de la política de migración (2) se estudia la influencia de los criterios de selección y aceptación de individuos que migran entre las islas.

Los resultados, en ambos casos, confirman nuestra hipótesis sobre la utilización de los dEA como herramientas adecuadas para enfrentar problemas dinámicos reales. Lo cual se logra mediante la capacidad de equilibrar el comportamiento del algoritmo para favorecer la búsqueda de soluciones óptimas o la reacción rápida a los cambios del problema. Esto es un resultado clave para el diseño

e implementación de nuevos algoritmos teniendo en cuenta que los problemas dinámicos reales pueden cambiar siguiendo distintos patrones de comportamiento dinámico a lo largo del tiempo.

2.1.3.2 Esquemas de memoria global para EDO

Diferente a usar múltiples poblaciones, otro método muy común en EDO consiste en usar algún tipo de *memoria* para almacenar información de escenarios anteriores y utilizarla para adaptar la búsqueda después de un cambio (Branke 1999; Yang 2007). La Figura 5 presenta una taxonomía de las distintas variantes de memoria que existen en Optimización Dinámica, que examinamos con mayor detalle (Bravo, Luque, and Alba 2016).

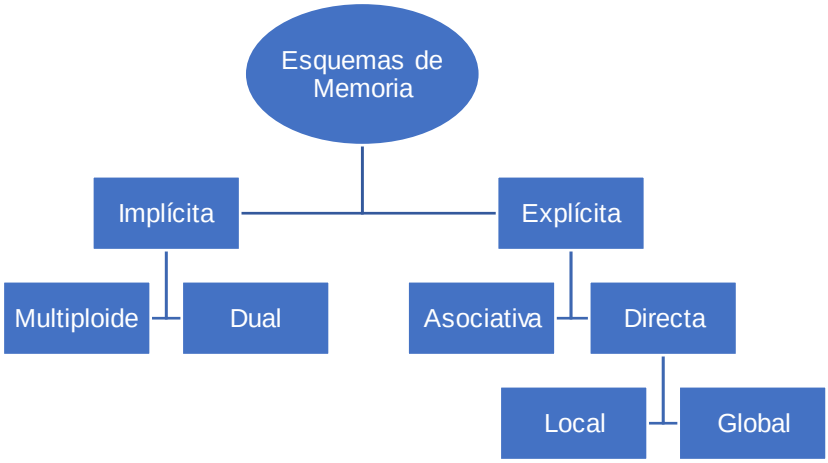


Figura 5. Taxonomía de esquemas de memoria en EDO.

Actualmente, muchos de los métodos que se proponen en EDO integran algún esquema de memoria (Brest et al. 2009; Halder, Das, and Maity 2013; Lepagnot et al. 2013; K. Trojanowski and Michalewicz 1999; Yang 2008; Yang and Li 2010), siendo uno de los esquemas más intuitivos y populares la *memoria global* (Simões and Costa 2011; Mavrovouniotis and Yang 2012; Yong and Wenjian 2010; Zhu, Luo, and Yue 2014). Este esquema consiste en usar una memoria *explícita* para almacenar directamente las buenas soluciones encontradas (posibles óptimos después de un cambio). Usar memoria global no sólo es efectiva para resolver DOP recurrentes (donde escenarios similares se repiten), sino también como mecanismo para introducir diversidad después de cada cambio.

Diseñar esquemas de memoria global para DOP requiere tomar decisiones sobre dos cuestiones de diseño clave: el *Reemplazo* y la *Recuperación*. Por un lado, es necesario establecer una estrategia de reemplazar las buenas soluciones encontradas, una vez que se alcance la capacidad de la memoria (debido a limitaciones físicas de almacenamiento). Las variantes actuales de reemplazo consideran uno o varios criterios (Eggermont et al. 2001; Shengxiang Yang 2005; Simões and Costa 2007; Woldesenbet and Yen 2009):

1. La *antigüedad* de las soluciones en la memoria.
2. La *calidad* de las soluciones dada la función objetivo.
3. La contribución de cada solución a la *diversidad* de la memoria.

Estas estrategias fueron inicialmente mencionadas en (Branke 1999), pero solo se analizaron variantes del criterio de diversidad (por ejemplo, *similitud* entre los individuos). Branke también mencionó la posibilidad de combinar más de un criterio a la vez, pero no lo analizó; señalando la dificultad de encontrar un balance equilibrado en este caso. Por el contrario, los esquemas de memoria más recientes usan estrategias combinadas (Lepagnot et al. 2013; Zhu, Luo, and Li 2011). Estas parecen ser más robustas ante distintos comportamientos dinámicos (Nguyen, Yang, and Branke 2012). Sin embargo, hasta donde conocemos, no existe un análisis integral del rendimiento las distintas variantes.

Por otro lado, las estrategias de *Reemplazo*, o cómo las soluciones son recuperadas de la memoria y reinsertadas en la población, suelen ser de dos tipos (Branke 1999; Brest et al. 2009; Wang et al. 2012): *Merging* y *Tracking*. En el primer caso, las soluciones de la memoria se *mezclan* con las soluciones de la población y se seleccionan los mejores individuos, o sea, las soluciones de la memoria reemplazan a individuos de la población (por ejemplo, (Branke 1999; Wang et al. 2012). Con la estrategia *Tracking* las soluciones de la memoria son recuperadas y re-exploradas de forma independiente, generando soluciones en una vecindad cercana a estas. El propósito aquí es evaluar si el nuevo óptimo es una variación de las soluciones de la memoria en el espacio de búsqueda (Brest et al. 2009; Lepagnot et al. 2013).

En resumen, los esquemas de memoria global presentan cuestiones de diseño claves, tales como ¿qué criterios seguir para actualizar la memoria? o ¿cómo recuperar las soluciones de la memoria? (RQ2). Nuevamente, no existe un análisis sobre las fortalezas y debilidades de cada implementación para distintos tipos de DOP (Branke 1999; Yang 2007). La tercera publicación realizada (3) durante esta investigación aborda esta cuestión (véase Sección 2.3). En concreto presenta un análisis integral de las técnicas que usan *memoria global* basado en una selección representativa de algoritmos y tipos de DOP. Los resultados obtenidos permiten construir mejores métodos basados en memoria, que son muy usuales en los trabajos más recientes (Simões and Costa 2011; Mavrovouniotis and Yang 2012; Yong and Wenjian 2010; Zhu, Luo, and Yue 2014).

2.2 Desarrollo de fundamentos teóricos en EDO

El desarrollo de fundamentos teóricos ha sido una tarea difícil para los investigadores en EDO, ya que sólo unos pocos estudios se han realizado a lo largo de los años (Stanhope and Daida 1998; Branke and Wang 2003; Droste 2004; Arnold and Beyer 2002; Zheng, Li, and Hu 2006; Rohlfshagen, Lehre, and Yao

2009; Oliveto and Zarges 2015). La falta de resultados en esta importante área se debe a la dificultad de caracterizar el efecto de los cambios del problema en el comportamiento del algoritmo.

Por ejemplo, en problemas de optimización estática es posible comparar la *velocidad de convergencia* entre dos variantes de EA por el *método de selección* utilizado (Goldberg and Deb 1991). Los autores han derivado modelos teóricos de *presión de selección* para varios métodos de selección analizando la dinámica de la población a lo largo de la ejecución (Goldberg and Deb 1991; Miller and Goldberg 1995; Popovici and De Jong 2003; Okabe 2007). Tales resultados no pueden extenderse directamente para DOP, ya que el espacio de búsqueda también puede cambiar de muchas maneras (por ejemplo, con mayor o menor severidad, con distinto periodo de cambio, ...) y, por tanto, el óptimo.

Una de las métricas usadas para medir la presión de selección en EA es el tiempo de absorción o *takeover time* (Goldberg and Deb 1991), que carece de una definición clara en EDO a pesar de su importancia. En las siguientes secciones presentamos algunas nociones básicas relacionadas con esta métrica, que nos servirán luego establecer un modelo de convergencia en EDO (Objetivo 3).

2.2.1 Curvas de crecimiento y Takeover time

Los modelos teóricos de *takeover time*, permiten describir el comportamiento de un algoritmo evolutivo. Esta métrica se basa en las *curvas de crecimiento* inducidas por un método de selección en un EA. La curva de crecimiento es una función $P_t: ℵ \rightarrow (0,1]$ que asigna la proporción de copias de la mejor solución en la población al paso de generación. Se asume la existencia de una única copia de la mejor solución en la población inicial ($P_0 = 1/n$, para un tamaño de población *n*). Como resultado, en cada generación el número de copias de la mejor solución actual crece. El número de generaciones que toma al algoritmo rellenar completamente la población de copias de la mejor solución es el *takeover time* estándar, y se define formalmente a continuación:

Definición 3 (*Takeover time* estándar): Se define como *takeover time* estándar, y se denota t^*, al valor:

$$t^* = min\{t: P_t = 1\} \quad (3)$$

donde $P_t: ℵ \rightarrow (0,1]$ es una función que define la curva de crecimiento inducida por un método de selección.
■

Como se señaló anteriormente, la definición de *takeover time* es importante en sí misma para construir el cuerpo de conocimiento teórico en EDO. Pero también puede utilizarse en aplicaciones reales, mediante modelos matemáticos que permiten calcular los valores de esta métrica a partir de los parámetros del EA

(por ejemplo, el *tamaño de la población* o el *método de selección* utilizado). En la siguiente sección se presenta un modelo de *takeover time* para problemas estáticos, usado luego en el cálculo de *takeover time* para problemas dinámicos.

2.2.2 Modelo estándar de Takeover Time

Se han propuesto varios modelos matemáticos para estimar el takeover time en problemas estáticos (Goldberg and Deb 1991; Miller and Goldberg 1995). Por ejemplo, considerando *selección por torneos*, uno de los métodos de selección más utilizados en la literatura, por lo que muchos más trabajos pueden beneficiarse directamente de los resultados aquí obtenidos.

En problemas estáticos, la curva de crecimiento teórica producida por el método de selección por torneos se define como (Goldberg and Deb 1991):

$$P_t = 1 - (1 - P_0)^{s^t} \tag{4}$$

donde s es el número de soluciones escogidas en cada proceso de selección (tamaño del torneo), P_0 es la proporción de copias de la mejor solución en la población inicial y t es el número de generación. Esta ecuación, cuando se resuelve en función de t considerando una única copia de la mejor solución en la población inicial, i.e., $P_0 = 1/n$, deriva en:

$$t^* = \frac{1}{ln\, s}[ln\, n\ +\ ln\,(ln\, n)] \tag{5}$$

donde *n* es el tamaño de población.

De nuevo, esta ecuación se refiere como modelo de *takeover time* estándar, en alusión al modelo propuesto inicialmente para problemas estáticos. De hecho, éste sólo representa las dinámicas o comportamiento del EA, sin considerar posibles dinámicas de cambio del problema, como sí ocurren en DOP. En la siguiente sección se muestran las curvas de crecimiento en un DOP, de cuyo análisis surgen nuestras propuestas para la definición y cálculo del *takeover time* en DOP presentadas luego en el apartado de resultados.

2.2.3 *Takeover time* en DOP

El modelo de *takeover time* estándar sirve para caracterizar la convergencia de algoritmos evolutivos en problemas estáticos. Sin embargo, su base en la "mejor solución" de la población no permite extender el concepto directamente a DOP, por los continuos cambios del óptimo en el tiempo (RQ3 y RQ4). Sin embargo, en un DOP, la curva de crecimiento parece incrementarse en cada intervalo estacionario, pero cae siempre después de un cambio en el problema, mostrando diferentes comportamientos (ver Figura 6).

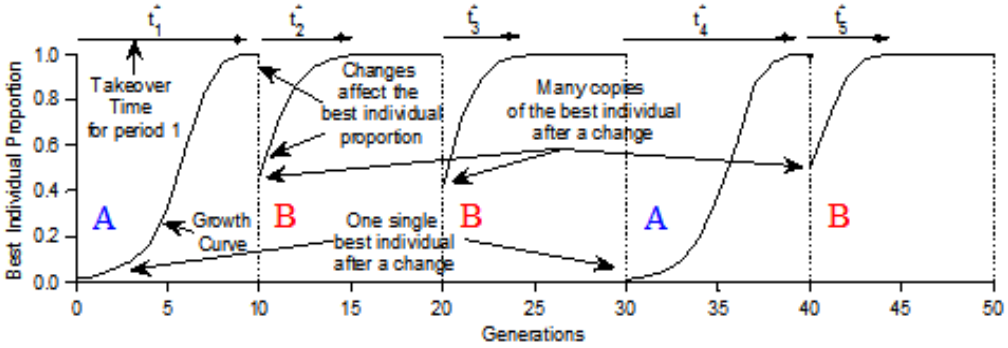


Figura 6. Curva de crecimiento en un DOP, descubriendo diferentes valores de *takeover time* a lo largo de la ejecución.

En la Figura 6 el lector puede distinguir dos escenarios diferentes, *A* y *B*, después de que se produce un cambio en el problema (cada 10 generaciones):

Escenario A. Aparece una nueva clase de mejor *fitness*, o

Escenario B. La mejor clase de fitness actual se mantiene después del cambio en el problema

Cada uno de estos dos comportamientos influye de forma diferente en la población de soluciones después de un cambio y por tanto en la curva de crecimiento. Consecuentemente, el desarrollo de nuestras contribuciones en las publicaciones (4) y (5)**,** relacionadas con la definición y cálculo del *takeover time* en DOP (RQ3 y RQ4), se derivan de analizar la probabilidad de ocurrencia de estos dos escenarios (véase Sección 3.4 y Sección 3.5).

Hay razones importantes que justifican el interés en la métrica del takeover time. Al contrario de lo que un análisis superficial podría decir, el takeover time no solo es interesante en teoría, sino que también puede servir de guía al algoritmo para la toma de decisiones y nos puede permitir crear técnicas adaptativas o auto-adaptativas. De hecho, esta idea ha atraído ya la atención de los investigadores para mejorar el rendimiento de EA en problemas estáticos (Osorio, Alba, and Luque 2013; Luque and Alba 2013), y es también útil para abordar diferentes dinámicas de cambio en DOP. Por lo tanto, además de contribuir a crear un cuerpo de conocimiento en EDO, también el takeover time puede ayudar a construir técnicas mejoradas para DOP (RQ5).

2.3 Validación y aplicación a problemas reales

La tercera gran área de desafíos en EDO tiene que ver con la validación de los resultados metodológicos y de fundamentos, así como su aplicación a problemas reales. Entre las problemáticas que encontramos en esta dirección se

incluyen: la búsqueda de métricas efectivas para evaluar y comparar el rendimiento de las técnicas EDO propuestas, el diseño de plataformas con DOP académicos que simulan distintos comportamientos dinámicos en entornos controlados, y la aplicación práctica en casos o escenarios DOP reales. Las secciones a continuación introducen los avances principales y perspectivas actuales en estas tres direcciones, respectivamente.

2.3.1 Métricas de rendimiento en EDO

Otra cuestión importante en EDO es cómo evaluar y comparar dos métodos o enfoques algorítmicos diferentes. En este dominio de problemas la mejor solución encontrada en cada instante no necesariamente describe el rendimiento del algoritmo, debido a que el óptimo cambia con frecuencia. Por ello, los investigadores han propuesto un conjunto de métricas de rendimiento específicas para la EDO.

Las métricas existentes se pueden agrupar en dos categorías: las basadas en *fitness* y las basadas en comportamiento (Ben-Romdhane, Alba, and Krichen 2013). Las primeras evalúan el rendimiento del algoritmo en función de la cercanía de las soluciones encontradas al óptimo global en cada momento. Por otro lado, las técnicas basadas en comportamiento se fijan en aspectos propios del algoritmo tales como la capacidad para mantener diversidad, para adaptarse rápidamente a cambios o la degradación del *fitness* a lo largo de la ejecución. Las métricas más comunes son métricas basadas en *fitness*, entre ellas las más importantes son *Best of Generation* (BOG), *Offline error*, *Accuracy* y *Area Between the Curve* (ABC). A continuación se describen brevemente estas métricas.

- **Best of Generation (**BOG**)**: Es un método directo de medir el rendimiento del algoritmo a lo largo de toda la ejecución, promediando el *fitness* de la mejor solución en cada generación, es decir:

$$BOG = \frac{1}{N}\sum_{i=1}^{N} f(x_i{}^*) \tag{6}$$

  donde $x_t{}^*$ es la mejor solución en cada generación y N es el número total de generaciones.

- **Offline error**: Es similar al BOG, pero sólo considera la mejor solución en cada periodo estacionario, promediando los valores obtenidos a lo largo de todos los cambios sobre el total de cambios:

$$Off_{error} = \frac{1}{N}\sum_{t=1}^{N} \left| f\left(x_t^{best}\right) - f\left(x_t^{opt}\right) \right| \tag{7}$$

donde N es el número total de cambios, $x_t{}^{best}$ es la mejor solución encontrada justo antes de cada cambio del problema y $opt\ {}^t$es el óptimo para el periodo *t*.

- **Accuracy** (*Acc*): Consiste en el promedio del valor de error relativo normalizado a lo largo de la ejecución, es decir:

$$Acc = \frac{1}{N}\sum_{t=1}^{N}\frac{f\left(x_t^{best}\right) - F_t^{\min}}{F_t^{\max} - F_t^{\min}} \tag{8}$$

donde N es el número de generaciones y $F_t{}^{max}$y $F_t{}^{min}$ son los valores de fitness máximo y mínimo, respectivamente.

- **Area Below the Curve** (ABC): Calcula la integral definida por otra métrica *p*, como por ejemplo el BOG, a lo largo de toda la ejecución.

$$ABC_p^A = \frac{1}{N}\int_1^N p_A\left(x\right)dx \tag{9}$$

Cada una de estas métricas tiene sus ventajas y desventajas (Nguyen et al. 2013). *BOG* y Off_{error} son métodos directos que miden el rendimiento del algoritmo durante toda la ejecución; pero no permiten comparar dos algoritmos para ver si las diferencias son estadísticamente significativas. Una alternativa es calcular el valor medio durante toda la ejecución (mean BOG), pero el resultado puede estar sesgado por la diferencia de fitness de los diferentes escenarios. El *Acc* intenta resolver esta limitante normalizando el valor de fitness en cada periodo de cambio del problema, pero requiere información acerca de la mejor y la peor solución del espacio de búsqueda. El *ABC* adopta las curvas de performance de las otras métricas e intenta obtener un valor cuantitativo y una forma de comparar estadísticamente dos algoritmos, mediante el área entre las curvas.

En general, no existe una métrica que caracterice con total certeza el comportamiento de un DOP, y ello depende en gran medida del criterio de optimalidad establecido. En consecuencia, es muy común en EDO el uso de diferentes métricas para analizar los resultados obtenidos. Así mismo, es necesario promediar los resultados tras múltiples ejecuciones de un algoritmo y realizar un estudio de significación estadística para validar los resultados. La Figura 7 ilustra cómo se promedian los resultados tras *n* ejecuciones de un algoritmo que resuelve un problema a lo largo de *k* cambios.

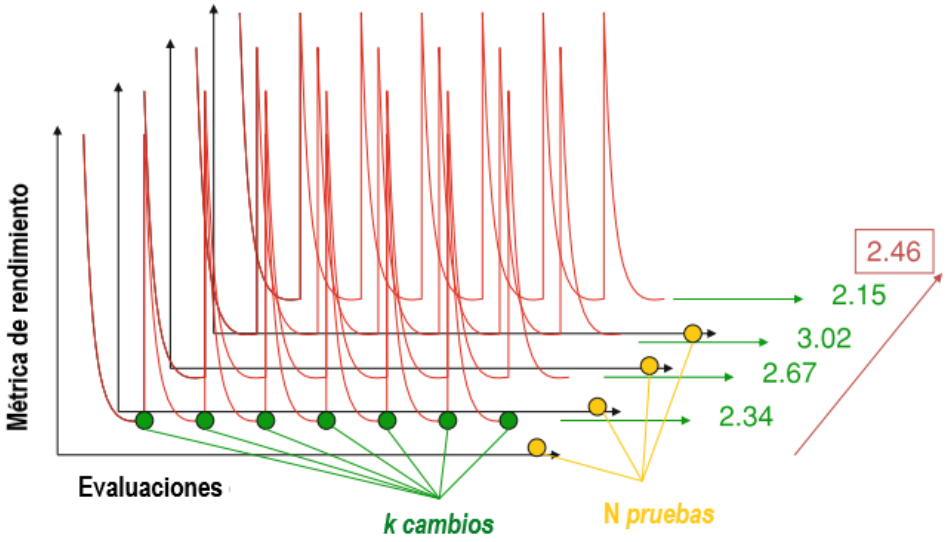


Figura 7. Método usual para el cálculo del rendimiento de un algoritmo en EDO. Los círculos representan el *fitness* de la mejor solución encontrada antes de un nuevo cambio en el problema (Cruz, González, and Pelta 2011).

En los trabajos publicados en esta tesis se sigue la siguiente metodología para comparar los algoritmos. En primer lugar, se usa el test de Kolmogorov-Smirnov (Lopes 2011) para verificar los datos siguen una distribución Normal. Si todos los resultados satisfacen el test de Normalidad entonces usamos el test de ANOVA para comparar los promedios de valores; si no, usamos el test de Kruskal-Wallis para comparar las medianas. En cada caso, buscamos un nivel de confianza superior al 95% o 99%. Para más información sobre métodos de comparación de algoritmos evolutivos y optimización se puede ver (Cruz, González, and Pelta 2011; García et al. 2008).

2.3.2 Plataformas de prueba y Problemas académicos

Otro aspecto importante en EDO ha sido el desarrollo de problemas académicos y plataformas de pruebas. Estas plataformas permiten evaluar el rendimiento de las técnicas desarrolladas, controlando diferentes aspectos de los DOP tales como la complejidad del espacio de búsqueda o las dinámicas de cambio.

Un conjunto de problemas de prueba comúnmente utilizados en EDO son las funciones construidas con el Dynamic Benchmark Generator (GDBG) (Li and Yang 2008a). Este define seis funciones (F1-F6) con distintas topologías, compuestas por picos, que implementan distintas dinámicas de cambio. Por ejemplo, los picos pueden cambiar aleatoriamente su altura, ancho y posición en el *fitness landscape* (véase Figura 8).

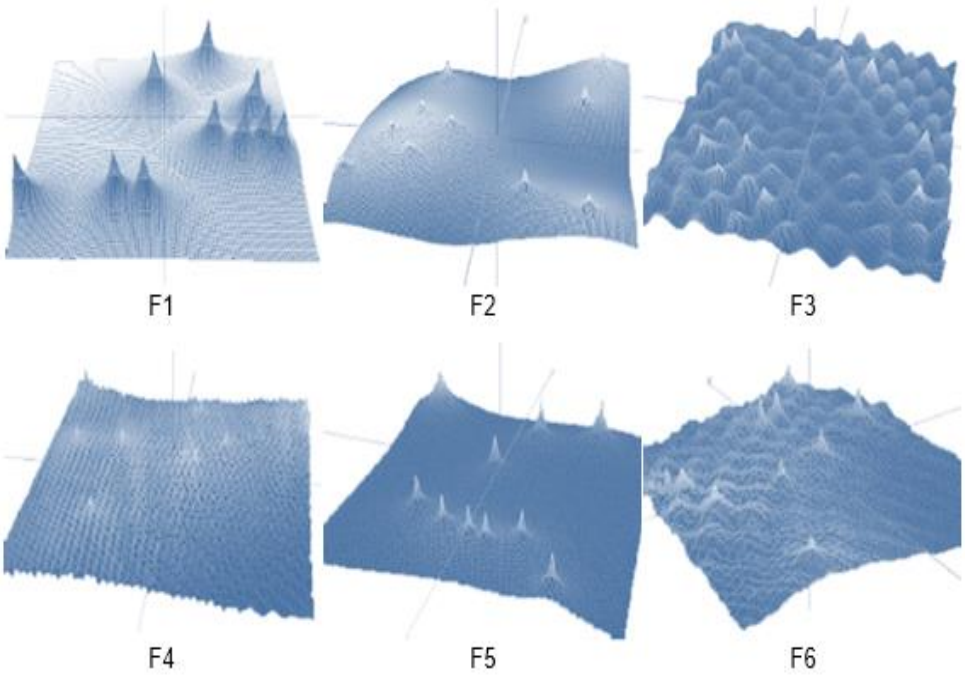


Figura 8. Las seis funciones de la plataforma GDBG (Li and Yang 2008a).

De las funciones que se muestran en la Figura 8, F1 y F2 consisten en un pequeño número de óptimos, que son exactamente los picos. Este es un escenario fácil a priori para los enfoques basados en la memoria, una vez que puedan descubrir las posiciones de los picos en el espacio de búsqueda. F3 y F4 son especialmente difíciles ya que añaden muchos óptimos locales difícil de encontrar. F3 tiene varios óptimos locales en regiones planas del paisaje, resultando también en un escenario de engaño para el algoritmo. La función F4, cuando se usa una dimensión baja del problema, tiene muchos óptimos locales similares cerca de cada pico, dificultando que el algoritmo encuentre el óptimo global. F5 también es altamente multimodal, pero sus óptimos locales son más como un pequeño ruido en el paisaje, por lo que es más fácil para el algoritmo escapar de ellos. Finalmente, la función F6 mezcla diferentes características juntas.

Además de la topología del espacio de búsqueda, las plataformas de prueba incorporan dinamismo. El GDBG puede generar pequeños o grandes cambios, cambios aleatorios, cambios caóticos, o recurrentes, con o sin ruido, y variar el número de óptimos o picos en el espacio de búsqueda.

En general, las plataformas de prueba intentan simular un problema DOP real en un entorno de cómputo controlado. Sin embargo, los trabajos basados en plataformas de prueba y problemas académicos suelen realizar un conjunto de asunciones sobre los DOP que pudieran no cumplirse en un escenario real. Esto incluye los objetivos de optimización, las restricciones, la visibilidad y detectabilidad de las soluciones, entre otros (Cheng and Yang 2010; Li-Ning Xing et al.

2011; Cheng, Yang, and Wang 2012). De aquí que se promueven cada vez más las aplicaciones de estas técnicas a problemas y casos reales de la ciudad. La siguiente sección aborda esta dirección con más detalle.

2.3.3 Casos de estudio en Movilidad Inteligente: Aplicaciones

En contraposición a lo visto en la sección anterior sobre DOP académicos, también se promueven las aplicaciones a problemas dinámicos del mundo real. Algunos autores han venido enfrentando este desafío con éxito. Ejemplos incluyen aplicaciones en biomedicina, minería de datos y procesamiento de imagen, entre otros (Richter 2009; Cruz, González, and Pelta 2011; Yang, Jiang, and Nguyen 2013; Alba, Nakib, and Siarry 2013).

Además, muchos dominios emergentes podrían beneficiarse de las técnicas EDO. Algunos ejemplos son los problemas de optimización en la Movilidad Inteligente, como la Gestión Inteligente de Cadenas de Suministro (que permite servicios eficientes y ecológicos bajo requisitos de operación variable), sistemas de semáforos (adaptación a intensidades de vehículos reales) y sistemas de *Ride-sharing* (maximizar la reutilización del vehículo bajo constante demanda de nuevas solicitudes de viaje), entre otros.

Estos problemas exigen cada día más atención por parte de investigadores y gestores del tráfico. Las frecuentes congestiones y atascos de tráfico son claras evidencias de la insuficiencia de las soluciones estáticas fijas existentes. Aquí, las técnicas de EDO podrían ayudar en la solución de estos problemas, cuyas condiciones y características son inherentemente dependientes del tiempo, de una manera dinámica.

La última publicación que se incluye en esta tesis (6) es un primer paso para abordar el tercer gran desafío en el campo de la optimización dinámica evolutiva. En particular, la aplicación de técnicas de computación evolutiva a problemas reales de nuestras ciudades (RQ 6). Esta investigación está más relacionada con el trabajo futuro, que en gran medida ya se ha iniciado para seguir trabajando en las líneas de impacto claras de aplicación real que se derivan de esta tesis. Por tal razón, no trata el problema directamente como dinámico, sino que se enfoca en entender el problema y desarrollar algoritmos bases de optimización evolutivas que después serán mejorados con las técnicas aquí desarrolladas.

En conclusión de este gran apartado de Preliminares (Sección 2), todas las publicaciones incluidas en esta tesis doctoral tienen un nexo común: a lo largo de ellas se diseñan, implementan y evalúan nuevos algoritmos evolutivos para problemas de optimización dinámicos. Los aspectos fundamentales relacionados con los resultados alcanzados durante el periodo de investigación de esta tesis doctoral se discuten en la siguiente sección.

3 RESUMEN DE RESULTADOS

En esta sección presentamos un resumen de los resultados de cada una de las publicaciones científicas aportadas en esta tesis. De hecho, no mostramos todos los detalles de la publicación sino una presentación y breve discusión de una selección de ellos. Para más detalles y otros resultados en cada caso se recomienda ir al artículo en cuestión siguiendo la referencia correspondiente al inicio de cada sección.

3.1 Influencia del periodo de migración en el diseño de algoritmos paralelos distribuidos para problemas dinámicos

Y. Bravo, G. Luque, E. Alba
Learning and Intelligent Optimization (LION), 2012
DOI: 10.1007/978-3-642-34413-8_25

Este estudio muestra cómo ajustar el periodo de migración, un parámetro clave de la política de migración en modelos paralelos distribuidos, al enfrentar problemas dinámicos (RQ1). Con tal propósito usamos un conjunto considerable de DOP, incluyendo tanto unimodales (con una única solución óptima) como multimodales (con muchas soluciones óptimas), y también con codificaciones diferentes. Estudiamos un *periodo de cambio* fijo ($\tau = 50$ generaciones) pero con distintos grados de *severidad de cambio*. Entonces, probamos distintos *periodos* de migración para evaluar el rendimiento del algoritmo en cada problema y evaluamos la métrica ABC_{ACC} (ver Sección 2.3.1). Altos valores de esta métrica (ABC_{ACC}) denotan mejor adaptación del algoritmo a los cambios del problema. Los resultados obtenidos se muestran en la Figura 9.

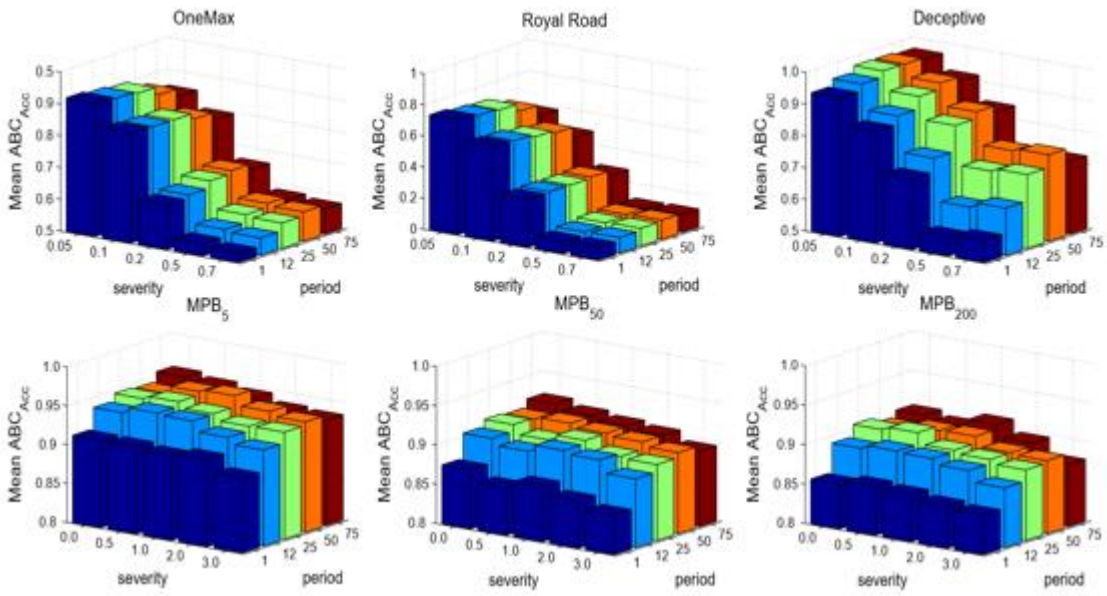


Figura 9. Influencia del periodo de migración en el rendimiento del dEA en problemas dinámicos con diferentes grados de severidad.

Los resultados en la Figura 9 muestran que es beneficioso usar un periodo de migración bajo para resolver problemas unimodales con cambios pequeños (es-

cenarios sencillos). Sin embargo, un período de migración alto resulta más robusto para enfrentarse a un gran número de severidades dinámicas en todos los problemas resueltos. Esto se debe a que el desacoplamiento de las islas permite mejorar la diversidad y la especiación de la población. También cabe destacar que la influencia del periodo de migración es menor para problemas masivamente multimodales (por ejemplo, MPB_{200}).

Además, un examen detenido de los resultados numéricos (Tabla 1) muestra que migrar como respuesta a un cambio en el problema es muy efectivo como mecanismo para adaptarse a los cambios del problema. Se puede notar que el mejor rendimiento del *dEA* (resultados marcados con *) se alcanza cuando el valor del periodo de migración coincide con el valor del periodo de cambio del problema ($\zeta = \tau = 50$).

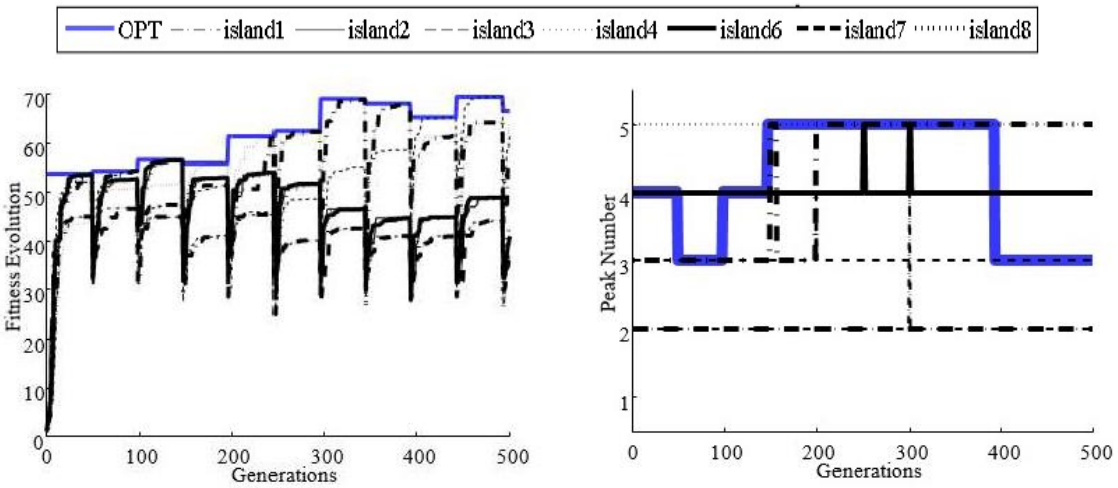
Tabla 1. *ABC_{Acc}* medio calculado para diferentes periodos de migración del dEA y varios DOP con diferentes severidades de cambio. La marca * indica la mejor configuración y las negritas que no hay diferencias estadísticamente significativas con respecto a esta.

ζ	ρ = 0.05	ρ = 0.1	ρ = 0.2	ρ = 0.5	ρ = 0.7	ρ = 0.0	ρ = 0.5	ρ = 1.0	ρ = 2.0	ρ = 3.0
	Onemax					MPB5				
1	**0.919***	**0.845***	0.639	0.539	0.528	0.912	0.911	0.908	0.912	0.896
12	0.912	**0.843**	0.649	0.561	0.554	0.939	**0.946**	0.945	**0.932**	0.923
25	0.905	**0.842**	0.671	0.585	0.579	**0.948**	**0.952**	0.945	**0.937**	**0.937**
50	0.879	0.824	**0.681***	**0.599***	**0.592***	0.946	**0.956***	**0.959***	**0.947***	**0.942***
75	0.852	0.800	0.664	0.588	0.579	**0.958***	**0.953**	0.947	**0.938**	**0.935**
	RoyalRoad					MPB50				
1	**0.734***	**0.601***	**0.306**	0.0758	0.0777	0.874	0.860	0.870	0.860	0.850
12	0.720	**0.599**	**0.313**	0.0944	0.098	**0.906**	0.898	**0.907**	**0.903**	0.887
25	0.711	**0.579**	**0.311**	0.110	0.114	**0.915**	0.903	**0.911**	**0.899**	**0.897**
50	0.661	0.562	**0.319***	**0.118***	**0.122***	**0.913**	**0.916**	**0.912**	**0.908***	**0.903***
75	0.589	0.491	0.286	0.111	0.114	**0.923***	**0.916***	**0.912***	**0.908**	**0.896**
	Deceptive					MPB200				
1	0.936	0.851	0.722	0.559	0.570	0.856	0.852	0.856	0.848	0.851
12	0.957	0.877	0.765	0.631	0.648	**0.898**	0.900	**0.894**	**0.887**	**0.879**
25	**0.977***	0.917	0.846	0.723	0.732	**0.901**	**0.901**	**0.900**	**0.894**	**0.882**
50	0.976	**0.937***	**0.867***	**0.764***	**0.772***	0.897	**0.910**	**0.908***	**0.900***	**0.894***
75	0.969	0.921	0.846	0.723	0.728	**0.913***	**0.914***	**0.897**	0.886	**0.888**

En esta publicación también ilustramos, por primera vez, la diversidad mejorada y la especiación natural de los modelos paralelos distribuidos, dos características muy favorables para diseñar nuevos algoritmos en optimización dinámica (véase la Figura 10).

La Figura 10 muestra la evolución del fitness y el óptimo del problema (multimodal) que está siendo explotado por cada isla en una ejecución del algoritmo. Aquí se puede ver cómo el modelo dEA es capaz de mantener la diversidad de la población y promover la especiación por la evolución independiente de las islas. El proceso de especiación consiste en un agrupamiento de individuos con

características similares (*especies*), debido al cruzamiento inducido por la estructuración de la población en islas.



a) Evolución del *fitness* ***b)*** Pico siendo explotado

Figura 10. Evolución del fitness (a) y pico siendo explotado (b) por isla en un dEA con alto periodo de migración.

En la evolución del fitness se puede observar que las curvas están espaciadas entre sí. Este comportamiento se corresponde con la capacidad del dEA para perseguir varios óptimos candidatos a la vez. Sin embargo, este comportamiento se logra con un periodo de migración medio alto. Si el periodo es bajo, lo que propicia un alto acoplamiento de las islas, el comportamiento es más parecido al de un EA secuencial donde la población persigue principalmente un único óptimo. Este último comportamiento tiene su utilidad como indicamos antes cuando el problema es unimodal y los cambios no son muy drásticos (severidad baja).

Esta investigación, a partir del análisis anterior y otros más expuestos en la publicación referenciada, descubre las fortalezas y limitaciones del dEA en EDO. Concretamente, caracterizamos la influencia del periodo de migración respecto a diferentes tipos de DOP, dando indicaciones de los valores que son recomendables atendiendo a las características del problema. De esta forma, validamos que los dEA son una herramienta muy favorable para el diseño de nuevos algoritmos evolutivos para problemas dinámicos, contribuyendo así al propósito general de esta tesis doctoral.

3.2 Influencia de los criterios de selección y reemplazo en modelos paralelos distribuidos para DOP

Y. Bravo, G. Luque, E. Alba
Distributed Computing and Artificial Intelligence (DCAI), 2013
DOI: 10.1007/978-3-319-00551-5_19

Este trabajo es continuación casi directa del anterior, donde seguimos analizando y caracterizando la influencia de las políticas de migración de los dEA a la hora de afrontar DOP (RQ1). A diferencia del estudio anterior donde nos centramos en el periodo de migración, ahora se analiza el rendimiento del modelo paralelo con diferentes criterios de selección y reemplazo de los individuos que migran. Aquí usamos la versión canónica del modelo paralelo distribuido *sin mutación*, para eliminar posible sesgo en los resultados. Además, consideramos tanto criterios basados en calidad (*fitness*) de las soluciones (por ejemplo, mejor o peor individuo) como en *distancia* (el individuo más cercano o más alejado del resto de miembros de una isla en el espacio genotípico de las soluciones), y también posibles combinaciones de valores de ambos criterios con selección aleatoria (véase Tabla 2). Igual que en el estudio anterior, usamos un conjunto amplio de problemas (*Onemax*, *Royal-Road*, *P-Peaks*, *MMDP*) con distintas severidades de cambio. Adicionalmente, consideramos diferentes tipos de cambio: cambios cíclicos (***Cy****clic*), cambios cíclicos con ruido (***Cy****clic with* ***N****oise*) y cambios aleatorios (***Ra****ndom*).

Tabla 2. Criterios de selección y reemplazo de migrantes.

Estrategia	**Descripción**
best-worst	Los buenos individuos reemplazan a los peores individuos
best-random	Los buenos individuos reemplazan a individuos aleatoriamente.
best-distant	Los buenos individuos reemplazan los individuos más distantes
rand-worst	Individuos aleatoriamente seleccionados reemplazan a los peores individuos
rand-distant	Individuos aleatoriamente seleccionados reemplazan a los más distantes
worst-worst	Los peores individuos reemplazan a los peores individuos
distant-distant	Los individuos más distantes y reemplazan a los más distantes

En este estudio, analizamos la influencia de las políticas de migración tanto en la calidad de las soluciones, como en mantener la diversidad (nivel medio-alto de acoplamiento) de la población. La Tabla 3 muestra los resultados obtenidos cuando se estudió la calidad de las soluciones con distintas políticas de migración ante distintos tipos de DOP. Dado que estamos interesados en el efecto de las políticas de migración, hemos agrupado los resultados obtenidos con diferentes severidades de cambio, calculando el número de experimentos en los que la política de migración utilizada es estadísticamente mejor que el resto (los valores

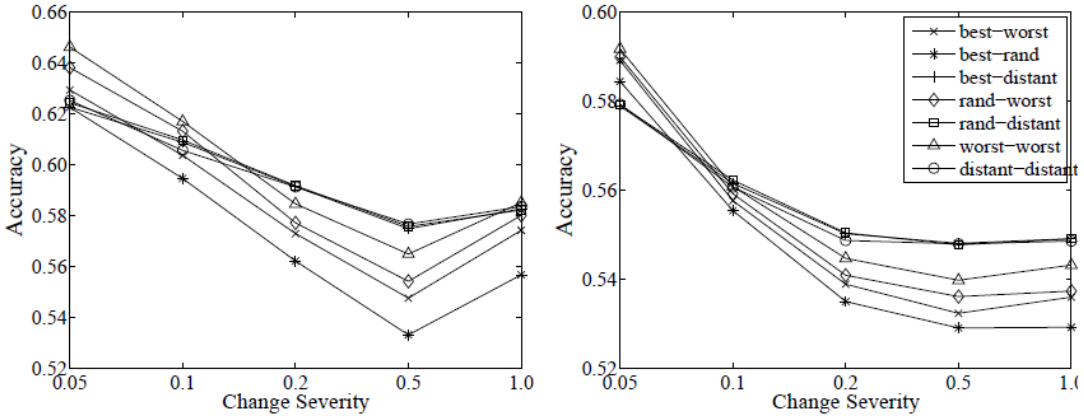
van de 0 a 5). Un mayor valor de esta métrica indica mejor capacidad de adaptación a DOP diferentes.

Por un lado (considerando la calidad de las soluciones encontradas), los resultados indican seleccionar el peor individuo, tanto para enviarlo de una isla a sus vecinos como para comparar y reemplazar por los individuos que migran (*worst-worst*). Esta política de migración es contra intuitiva en dEA para problemas estáticos, pero es la mejor en un 86,67 % de los DOP probados. La razón es que permite perseguir un mayor número de óptimos locales a la vez, ya que promueve la evolución independiente de las islas (*isolation*) formando bases de atracción para los individuos que migran. Este comportamiento impide así que la mejor solución domine al resto. De esta forma, el dEA puede perseguir los movimientos de los óptimos encontrados.

Tabla 3. Número de experimentos en los que la política de migración es estadísticamente mejor que el resto. Los mejores valores están en negritas.

Migration policy	Onemax			Royal-Road			P-Peaks			MMDP			Tot
	Cy	*CyN*	*Ra*	*Cy*	*CyN*	*Ra*	*Cy*	*CyN*	*Ra*	*Cy*	*CyN*	*Ra*	
best −worst	3	1	1	3	0	0	2	0	0	0	0	0	1
best −random	0	0	0	0	0	0	1	0	0	0	0	0	1
best −distant	0	**4**	**4**	0	**4**	**4**	0	0	0	0	0	0	1
rand −worst	**4**	3	2	3	2	2	3	1	1	0	0	0	1
rand −distant	1	**4**	**4**	0	**4**	**4**	0	0	0	0	0	0	1
worst −worst	**5**	**4**	2	**5**	**4**	2	**5**	**5**	**5**	**5**	**5**	**5**	5
distant −distant	0	3	**4**	0	**4**	**4**	0	0	0	0	0	0	1

No obstante, el buen rendimiento de las políticas de reemplazo basada en *fitness* (incluida *worst-worst*) está afectado por la severidad de cambio. La calidad de las soluciones encontradas disminuye en la medida que aumenta la severidad de cambio (véase Figura 11).



a) Onemax: Cambios cíclicos con ruido

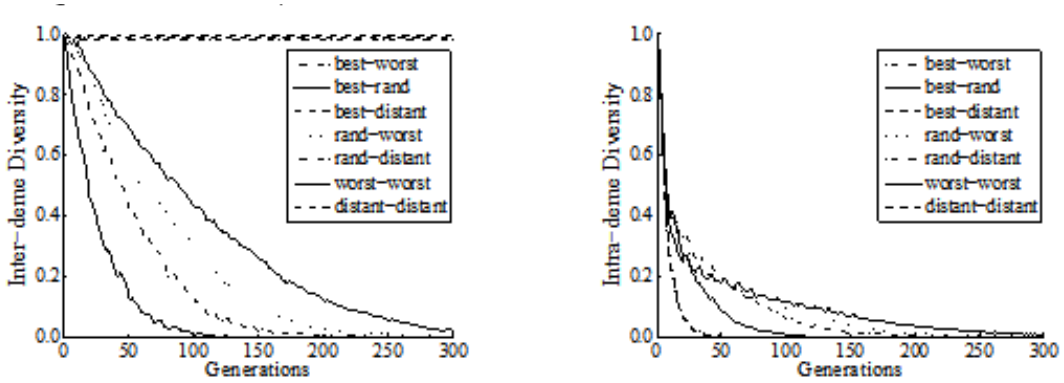
b) Onemax: Cambios aleatorios

Figura 11. Influencia de diferentes políticas de migración en el rendimiento del dEA para el problema Onemax cíclico con ruido (a) y con cambios aleatorios (b), considerando diferentes severidades de cambio.

Frente a DOP con severidad de cambio alta, los resultados de la Figura 11 indican que las estrategias de reemplazo basadas en distancia (*best-distant*, *rand-distant*, y *distant-distant*) permiten obtener soluciones de mayor calidad. Esto se debe a que tales estrategias promueven la diversidad global (bajo acoplamiento entre las islas) y la especiación. Una mayor diversidad global permite al dEA reaccionar y adaptarse a DOP con cambios abruptos y discontinuos, mientras que la especiación ayuda a perseguir soluciones distantes que son útiles para guiar la búsqueda tras cada cambio.

Para verificar estas observaciones también estudiamos la influencia de las políticas de migración en la diversidad de la población ante distintos tipos de DOP. Para ello calculamos la desviación estándar o STD en el espacio de Hamming de las soluciones para determinar la diversidad *genotípica* global (entre las soluciones de distintas islas o *inter-deme*) y local (entre las soluciones dentro de cada isla o *intra-deme*). La Figura 12 muestra los resultados obtenidos.

Efectivamente, las estrategias de reemplazo basadas en distancia resultan ser más efectivas para perseguir un mayor número de óptimos locales, debido a que promueven la diversidad global y, por tanto, la especiación de la población (ver el valor 1 de la curva en la Figura 12 a). Sin embargo, el continuo reemplazo de los individuos, por su distancia con respecto al resto de la subpoblación, promueve a largo plazo la pérdida de diversidad local. Como resultado, el algoritmo no puede perseguir los movimientos de las buenas soluciones encontradas. El buen rendimiento de estas estrategias que se observó antes (véase Tabla 3) se relaciona únicamente con los problemas unimodales, donde lo importante es una rápida adaptación a los cambios.



a) Diversidad global (inter-deme) ***b)*** Diversidad local (intra-deme)

Figura 12. Efecto de la política de migración sobre la diversidad en la población para el problema MMDP con modo de cambio aleatorio (Non-cyclic) y severidad $\rho = 0.5$.

En conclusión, este estudio confirma que los criterios de selección y reemplazo de individuos que migran entre las islas son una herramienta útil para ajustar el algoritmo a la hora de resolver problemas dinámicos. Concretamente, permiten al algoritmo equilibrar su funcionamiento entre **(1)** la búsqueda de soluciones óptimas para un escenario del problema y **(2)** la adaptación rápida a los cambios del problema. Un resultado que resalta nuestro interés en este parámetro de cara a diseñar e implementar mejores métodos EDO.

3.3 Esquemas de memoria global para el diseño de nuevas técnicas de optimización dinámica

Y. Bravo, G. Luque, E. Alba
Nat. Comput., 2015
DOI: 10.1007/s11047-015-9497-2

En las anteriores publicaciones se analizó una de las posibles estrategias usadas para adaptar los EA a la resolución de DOP. En esta publicación seguimos con la misma idea (seguimos en el campo de la Metodología según definimos en la introducción) pero abordamos el estudio de otra técnica diferente: el uso de la *memoria*. Concretamente abordamos el estudio integral de los métodos que usan *memoria global* para adaptarse a los cambios del problema, por ser estos los más intuitivos y fáciles de integrar con otros métodos. Con tal propósito, diseñamos un modelo genérico uniforme de memoria global, que toma como parámetros las estrategias de memoria a utilizar y el algoritmo evolutivo base (cuyo rendimiento se mejora con la estrategia de memoria seleccionada).

Con propósito de analizar las técnicas EDO basadas en memoria global, lo primero que hicimos fue introducir un conjunto de métricas para evaluar la posible contribución del uso de memoria (*Memory Update*, *Memory Accuracy*, *Memory Diversity* y *Memory Performance*), supliendo así una necesidad común en los estudios anteriores de otros autores, basados solo en las métricas definidas para los algoritmos (por ejemplo, *Offline Performance*). Por ejemplo, definimos cómo medir el *Memory Performance* o rendimiento de la memoria mediante la proporción de óptimos locales del problema almacenados en memoria, con respecto al total de óptimos globales y el tamaño de la memoria, i.e.:

$$Perf^t = \frac{{n_t}^*}{min(|M|, {n_t}^{opt})} \tag{10}$$

donde ${n_t}^*$ es el número de soluciones de la memoria que muestran picos en el espacio de búsqueda, ${n_t}^{opt}$ es el número total de picos, y |M| es la capacidad de la memoria. Valores bajos de esta métrica revelan la incapacidad del algoritmo para buscar picos o la incapacidad de la memoria para retenerlos. El conjunto completo de métricas se puede acceder en la publicación de referencia (Bravo, Luque, and Alba 2016).

Finalmente, realizamos un análisis combinando distintos criterios utilizados para seleccionar la solución a reemplazar: antigüedad (***A**ge*), calidad (***F**itness*) y contribución a la diversidad (***S**imilar*); con distintos criterios para recuperar las soluciones de la memoria (*Tracking* y *Merging*). Usamos las distintas estrategias de memoria (en todas sus combinaciones posibles) y tres algoritmos base con

características diferentes (*meGA*, *mePSO*, y *meLS*). Cada algoritmo resultante se probó con las seis funciones construidas con el generador de problemas *GDBG* (véase Sección 2.3.2), con topologías y comportamientos diferentes. Los resultados muestran las ventajas y las limitaciones de cada estrategia para optimización dinámica, así como el efecto que tiene el algoritmo base seleccionado y las características del problema en el rendimiento de la Memoria.

Tabla 4. Eficacia de cada estrategia de actualización de la memoria para tratar varios tipos

		A	A-F	A-S	A-F-S	A-S-F	F	F-A	F-S	F-A-S	F-S-A	S	S-A	S-F	S-A-F	S-F-A
meGA	*F1*	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
	F2	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	85.7%	85.7%	**100.0%**	**100.0%**	**100.0%**
	F3	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	71.4%	71.4%	**100.0%**	**100.0%**	**100.0%**
	F4	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
	F5	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
	F6	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
mePSO	*F1*	42.9%	57.1%	42.9%	57.1%	57.1%	42.9%	42.9%	42.9%	42.9%	42.9%	85.7%	85.7%	85.7%	85.7%	85.7%
	F2	57.1%	57.1%	57.1%	57.1%	57.1%	28.6%	42.9%	57.1%	42.9%	42.9%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
	F3	42.9%	85.7%	42.9%	85.7%	85.7%	71.4%	85.7%	71.4%	85.7%	85.7%	85.7%	85.7%	**100.0%**	**100.0%**	**100.0%**
	F4	57.1%	85.7%	57.1%	85.7%	85.7%	71.4%	71.4%	71.4%	71.4%	71.4%	85.7%	85.7%	85.7%	85.7%	85.7%
	F5	28.6%	71.4%	28.6%	71.4%	71.4%	57.1%	42.9%	42.9%	42.9%	42.9%	42.9%	42.9%	**100.0%**	**100.0%**	**100.0%**
	F6	57.1%	57.1%	57.1%	57.1%	57.1%	42.9%	28.6%	28.6%	28.6%	28.6%	71.4%	71.4%	85.7%	85.7%	85.7%
meLS	*F1*	14.3%	14.3%	14.3%	14.3%	14.3%	0.0%	0.0%	0.0%	0.0%	0.0%	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
	F2	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	71.4%	71.4%	**100.0%**	**100.0%**	**100.0%**
	F3	28.6%	57.1%	28.6%	57.1%	57.1%	28.6%	14.3%	28.6%	14.3%	14.3%	42.9%	42.9%	**100.0%**	**100.0%**	**100.0%**
	F4	0.0%	14.3%	0.0%	14.3%	14.3%	0.0%	0.0%	0.0%	0.0%	0.0%	85.7%	85.7%	**100.0%**	**100.0%**	**100.0%**
	F5	28.6%	28.6%	28.6%	28.6%	28.6%	0.0%	0.0%	0.0%	0.0%	0.0%	28.6%	28.6%	**100.0%**	**100.0%**	**100.0%**
	F6	42.9%	42.9%	42.9%	42.9%	42.9%	42.9%	42.9%	42.9%	42.9%	42.9%	85.7%	85.7%	**100.0%**	**100.0%**	**100.0%**
	Total	22.2%	31.7%	22.2%	31.7%	31.7%	21.4%	20.6%	21.4%	20.6%	20.6%	80.2%	80.2%	97.6%	97.6%	97.6%

Una primera conclusión sacada de los resultados de este estudio fue que la eficacia de la estrategia de reemplazo depende en gran medida de la estrategia de recuperación empleada. Cuando se usó la estrategia *Tracking*, la contribución

a la *diversidad* de la memoria fue la mejor elección en todos los casos (véase la Tabla 4), mientras que cuando se usó la estrategia *Merging* las estrategias de reemplazo mostraron mayor rendimiento (81.7%) considerando todas las combinaciones de algoritmos y funciones utilizadas.Por otro lado, cuando se usan estrategias de reemplazo que combinan dos o más criterios básicos combinados, el orden en que ellos se aplican es muy importante. Entre otros resultados, nuestro estudio revela que la contribución a la diversidad (*S*) de la memoria es más importante que la calidad (*F*) o la antigüedad (*A*) de las soluciones en la memoria (véase Tabla 4). Esto motiva a revisar algunos trabajos en la literatura que buscan, en primer lugar, aumentar la calidad de las soluciones de la memoria (Lepagnot et al. 2013).

En relación a la estrategia de recuperación, usar *Tracking* resultó significativamente mejor que usar *Merging* en un 73% de los problemas. Como caso particular, cuando se usó de base un Algoritmo Genético (*meGA*), ante DOP con un número pequeño de óptimos u óptimos recurrentes muy precisos (T5), usar *Merging* fue mejor en el 100% de los casos (véase Tabla 5). Esto se entiende porque el algoritmo tiene que dedicar tiempo a re-explorar estas soluciones de la memoria que vuelven a aparecer exactamente en la misma posición del espacio de busca, limitando así la búsqueda de otras buenas soluciones.

Tabla 5. Comparación entre *Merging* y *Tracking* usando el error offline. Para cada caso, se seleccionaron los resultados con la mejor estrategia de reemplazo. Los triángulos negros (blancos) indican que la estrategia *Merging* (*Tracking*) es mejor. El guion indica que las diferencias no son estadísticamente significativas.

	meGA						mePSO						meLS			
	F1	F2	F3	F4	F5	F6	F1	F2	F3	F4	F5	F6	F1	F2	F3	F4
T1	▲	▽	▽	▽	▽	▽	▽	▽	▽	—	▽	▽	▽	▽	▽	▽
T2	▲	▲	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽	▽
T3	▲	▽	▲	▽	▽	▲	▽	▽	▽	▲	▽	▽	▽	▽	▲	▽
T4	▲	▽	▽	▽	▽	▲	▽	▽	▽	▽	▽	▽	▲	▽	▽	▽
T5	▲	▲	▲	▲	▲	▲	▽	▽	▲	—	—	▲	▽	▽	▲	▲
T6	▲	▽	▽	▽	▽	▲	▲	▽	▽	—	▽	▽	▽	▽	▲	▽
T7	▲	▲	▲	▲	▽	▲	▽	▽	—	—	▽	▽	▽	▽	▲	▽

Las características del algoritmo base también deben tenerse en cuenta cuando se diseñan e implementan esquemas de memoria global para problemas dinámicos. Los resultados indican que usar *Tracking* con un algoritmo *lento* (Por ejemplo, *meGA*) puede afectar la exploración del espacio de búsqueda. Lo cual se expresa en un número muy reducido de actualizaciones de la memoria (véase Figura 13). Por el contrario, usar esta estrategia con un algoritmo *rápido* (Por

ejemplo, *mePSO*) puede incrementar el número de soluciones óptimos locales en la memoria, en lugar de óptimos globales.

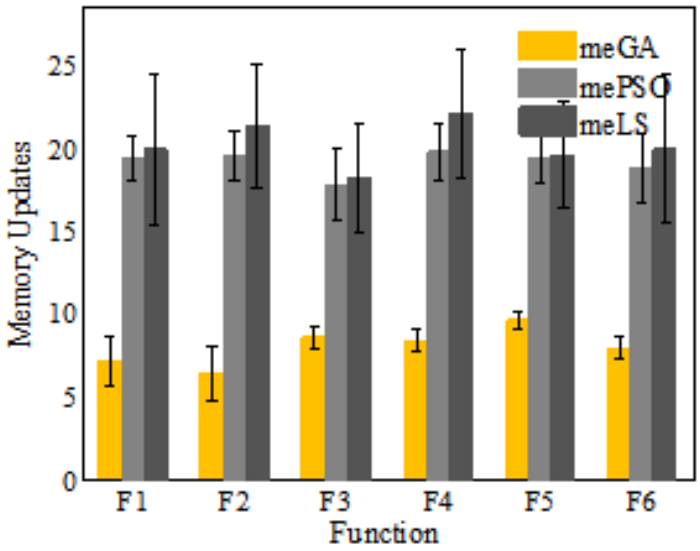


Figura 13. Cantidad media de actualizaciones de la memoria o *Memory Updates* para los diferentes algoritmos (sobre todos los tipos cambios y severidades).

Aquí hemos resumido algunos resultados del estudio al que se hacer referencia. En nuestro artículo se puede acceder a todos los parámetros del diseño experimental, así como encontrar otros resultados más completos y detallados.

3.4 Definición y cálculo de takeover time en algoritmos evolutivos para problemas de dinámicos

Y. Bravo, G. Luque, E. Alba
IEEE Symp. on Comput. Int., 2013
DOI: 10.1109/cidue.2013.6595768

Hasta aquí nos hemos dedicado al análisis experimental unificado de algunos de los métodos existentes en EDO (metodología); ahora nos enfocamos en el desarrollo de nuevos fundamentos. Concretamente, abordamos la caracterización de la convergencia de los EA en DOP mediante modelos de takeover time, como otra de las principales contribuciones de esta tesis (RQ 3-5). En esta primera publicación (de dos publicaciones realizadas sobre fundamentos) se introduce una definición de takeover time esperado para EDO. La misma deriva de la observación de escenarios característicos que afectan la población de soluciones después de un cambio (véase Sección 2.2.3). Específicamente, se define esta métrica como el valor esperado de takeover time en cada periodo estacionario, de acuerdo con la siguiente definición:

Definición 4 (*Takeover time* esperado): El *takeover time* esperado, denotado por $\hat{t}$, se define como:

$$\hat{t} = E[t_i{}^*] = \sum_{i=1}^{k} t_i{}^* \cdot p(t_i{}^*) \tag{11}$$

donde $t_i{}^*$ ($i = 1,2,3,\ldots,k,\ldots$) son los valores de takeover time para cada intervalo estacionario, que siguen una distribución de probabilidad $p(x)$.
■

Nosotros no nos limitamos a definir la métrica sino también desarrollamos un modelo ajustado para calcular su valor en algoritmos evolutivos que empleen *selección por torneo* y métodos que generan diversidad después de un cambio en el problema como mecanismo de adaptación. Por un lado, el método de selección por torneo es uno de los más usados en la literatura, y por tanto más trabajos pueden beneficiarse directamente de estos resultados. Por otro lado, generar diversidad después de cada cambio es uno de los métodos más simples para mejorar el rendimiento del EA en DOP (véase Sección 2.1.3). Este método consiste en reemplazar parte de la población por nuevas soluciones generadas aleatoriamente, con el fin de generar diversidad para adaptarse a los cambios en el problema.

El modelo propuesto toma en cuenta también características del DOP, específicamente la severidad de cambio. Además, se basa en los modelos estándares definidos en la literatura para problemas estáticos t^* (véase Sección 2.2.2). En este primer acercamiento, al modelar el takeover time solo consideramos periodos de cambio mayor que el valor del modelo estándar para problemas estáticos ($\tau \geq t^*$). Luego, en otra publicación eliminamos esta consideración para tener en cuenta cualquier periodo de cambio del problema.

Finalmente, el modelo desarrollado para cálculo del takeover time esperado se basa en la probabilidad de ocurrencia de los dos escenarios característicos presentados antes en la Sección 2.2.3. Después de un cambio en el problema, puede ocurrir que: (A) aparezca una nueva clase de mejor *fitness*, donde la proporción de individuos de mejor clase es mínima, esto es $P_0 = 1/n$, siendo n el tamaño de la población; o (B) que la mejor clase de fitness actual se mantenga después del cambio (véase Figura 5). Luego, el takeover time esperado se puede calcular mediante la Proposición como propone la Definición 5, validada luego de forma experimental mediante la propuesta de estimaciones precisas para sus parámetros:

Definición 5 (Cálculo del *Takeover time* en DOP): El *takeover time* en un DOP, definido como takeover time esperado y denotado por $\hat{t}$, se puede calcular como:

$$\hat{t} = \frac{\alpha \cdot {t_A}^* + (k - \alpha) \cdot {t_B}^*(\hat{P}_0)}{k} \tag{12}$$

donde $\alpha = 1,2,3,\ldots,k,\ldots$ es el número de veces que ocurre el comportamiento A a lo largo de k intervalos estacionarios, ${t_A}^*$ y ${t_B}^*$ son valores de takeover time obtenidos mediante la derivación del modelo estándar de takeover time (véase Sección 2.4). En los casos del escenario *A* ($P_0 = 1/n$), se usa directamente el modelo estándar) mientras que en los escenarios que siguen el esquema *B* ($P_0 > > 1/n$), el modelo estándar se resuelve para el valor real de P_0. $\hat{P}_0 \in [0,1]$ es la proporción inicial de buenas soluciones esperada en aquellos intervalos estacionarios donde ocurre *B*.

■

El modelo para el cálculo del takeover time propuesto en la Definición 5 se complementa en el estudio aquí referido con modelos concretos para calcular el valor de los parámetros α y $\hat{P}_0$. Dicho modelo parte de analizar que el parámetro α depende significativamente de la severidad de cambio del problema, mientras que el parámetro $\hat{P}_0$ está mayormente afectado por la *tasa de reemplazo* del algoritmo (m) al generar diversidad (proporción de nuevas soluciones insertadas).

Estimación 1 (Parámetros de la Definición 5 para $\tau \geq t^*$): Los parámetros de la Definición 5 pueden calcularse como sigue:

$$\alpha(\rho,m) = \begin{cases} 1, & m = 0 \\ h(\rho), & 0 < m < 1 \\ k, & m = 1 \end{cases} \quad (13)$$

donde $m \in [0,1]$ es la tasa de reemplazo, y h es una función de la severidad de cambio $\rho \in [0,1]$ definida parcialmente por la modalidad del problema como: $h(\rho) \approx \frac{k}{1+(k-1)e^{-b\rho^2}}$ si el problema es unimodal y $h(\rho) \approx k - (k-1)e^{-b\rho^2}$ en otro caso (multimodal), para algún valor b constante. Y $\hat{P}_0$ es una función dependiente de la tasa de reemplazo:

$$\hat{P}_0\,(m) \approx 1 - m^2 \quad (14)$$

donde *k* es el número de intervalos estacionarios.
■

Estas ecuaciones se obtuvieron mediante el modelado matemático usando funciones que describen el comportamiento del algoritmo y el ajuste de las curvas. Tanto el modelo matemático para el cálculo del takeover time, como los modelos concretos propuestos para sus parámetros, se validaron experimentalmente usando un conjunto representativo de problemas dinámicos, específicamente *OneMax*, *P-Peak*, *MMDP* (véase Sección 2.3.2). Y al igual que en los anteriores artículos, cada problema se probó con distintos valores de severidad de cambio, así como configuraciones del algoritmo (número de soluciones reemplazadas después de un cambio para generar diversidad). Además, con propósito de cuantificar la magnitud del error, se calculó también el error producido por el modelo de takeover estándar, que es considerado muy preciso en la literatura), denotado aquí como t^*_{Deb}. La Tabla 6 muestra los resultados obtenidos.

Los resultados de la Tabla 6 muestran que los errores producidos por nuestro modelo son de la misma magnitud que el alcanzado con el modelo estándar. Por lo tanto, el nuevo modelo propuesto puede ser usado para calcular con precisión el valor de *takeover time* esperado para EA que usen selección por torneo y un esquema "reactivo" para DOP.

Tabla 6. Error cuadrático medio para los diferentes valores de cambio de severidad (ρ) y ratio de reemplazo (m).

ρ	$m = 0.2$		$m = 0.4$		$m = 0.6$		$m = 0.8$	
OneMax	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$
0.1	2.129	2.366	2.169	2.212	2.199	2.388	2.186	1.986
0.2	1.957	2.036	2.019	1.850	2.072	2.232	2.065	2.084
0.3	1.227	1.982	1.359	2.132	1.452	2.268	1.561	2.454
0.4	1.586	1.084	3.208	1.040	3.920	1.001	3.948	0.967
0.5	1.877	0.951	1.812	0.943	1.661	0.867	1.568	0.883
P-Peaks	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$
0.1	2.655	0.821	2.513	0.775	2.366	0.780	2.137	0.746
0.2	1.064	0.788	1.108	0.804	1.016	0.717	0.964	0.718
0.3	0.760	0.780	0.733	0.740	0.763	0.748	0.739	0.743
0.4	0.662	0.763	0.690	0.753	0.691	0.752	0.728	0.778
0.5	0.676	0.796	0.690	0.771	0.645	0.702	0.716	0.770
MMDP	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$	$Err(t^*)$	$Err(t^*_{Deb})$
0.1	2.804	1.177	1.836	1.126	1.145	1.152	0.590	1.180
0.2	0.803	1.131	0.995	1.149	1.120	1.168	1.162	1.142
0.3	0.903	1.140	1.047	1.146	1.112	1.145	1.099	1.109
0.4	0.882	1.155	1.014	1.122	1.137	1.193	1.099	1.123
0.5	0.816	1.120	0.965	1.107	1.070	1.160	1.063	1.114

3.5 Mejores modelos de cálculo de *takeover time* y su aplicación al diseño de nuevos algoritmos para EDO

Y. Bravo, G. Luque, E. Alba
Applied Mathematics and Computation, 2015
DOI: 10.1016/j.amc.2014.10.107

Continuando con el desarrollo de fundamentos iniciado en la publicación anterior (véase RQ 3-5), en esta publicación se proponen nuevas estimaciones para los parámetros del cálculo de *takeover time* en EDO (véase Sección 3.4). El nuevo modelo resuelve las limitaciones del modelo anterior en relación a al periodo de cambio del problema, incluyendo casos no considerados antes. Este modelo mejorado prueba ser más eficiente que el anterior para un mayor número de problemas dinámicos. Además, se diseña una nueva técnica "reactiva" (que reacciona a los cambios del problema) y auto-adaptativa para optimización dinámica basado en los modelos matemáticos de *takeover time*.

El modelo de cálculo del *takeover time* inicial (publicación previa) sólo consideró periodos de cambio del problema mayores que el *takeover time* estándar ($\tau \geq t^*$). Este nuevo artículo elimina esta limitación, considerando no solo la severidad de cambio sino también cualquier periodo de cambio del DOP, incluyendo periodos de cambio menores que el takeover estándar ($\tau < t^*$). Como antes, se consideró el algoritmo evolutivo con selección por torneo y el método de generación de diversidad con tasa de reemplazo m.

Tomando como referencia el modelo propuesto en la Definición 5 para el cálculo del takeover en optimización dinámica, se proponen nuevas estimaciones para los parámetros involucrados en esta proposición:

Estimación 2 (Parámetros de la Definición 5 Revisados): Sean τ y ρ el periodo y la severidad de cambio de un DOP, respectivamente, k el número de iteraciones, t^*el valor de *takeover* estándar, y $m \in [0,1]$ la tasa de reemplazo del algoritmo; los parámetros de la Definición 5 pueden calcularse como:

$$\alpha(\rho, m, \tau) \approx k + (k - h_0) \cdot \frac{e^{-(1-m)\rho}}{1 + e^{(\frac{m}{2}t^* - \tau)}} \tag{15}$$

donde h_0 $(1 \leq h_0 \leq k)$ es un nuevo parámetro que denota un límite inferior para el valor de α definido por la Estimación 1 (ver Ecuación 13); y

$$\hat{P}_0\,(m, \tau) \approx (1 - m) \cdot (1 - e^{-d \cdot \tau / t^*}) \tag{16}$$

donde d es una constante que caracteriza la curva de crecimiento para $\tau < t^*$.

■

En esta publicación, además de proponer modelos teóricos de *takeover time*, también los usamos para diseñar nuevos algoritmos evolutivos para problemas dinámicos, específicamente, un método basado en el control de la diversidad. Siguiendo idea de procesamiento secuencial de la población (Goldberg 2002), el tiempo de convergencia t_c se definió como λ veces el valor de takeover $\hat{t}$, es decir:

$$t_c = \lambda \cdot \hat{t} \tag{17}$$

donde λ es un valor positivo dependiente del efecto de considerar operadores de variación (cruzamiento o mutación) en algoritmos evolutivos.

Entonces, se diseñó un nuevo algoritmo denominado $reEA_{\mu_m}$ que toma como parámetro el modelo de convergencia propuesto. Este algoritmo reacciona a los cambios del problema generando diversidad, al tiempo que ajusta automáticamente la presión de selección del algoritmo a partir del comportamiento dinámico del problema que está resolviendo.

El $reEA_{\mu_m}$ emplea el operador de mutación NUMO (siglas del inglés Non-uniform mutation operator) (Michalewicz 1996) que afecta menos a cada solución en la medida en que el tiempo de ejecución se aproxima a un valor máximo t_{max}. Concretamente, nuestro algoritmo aprende el valor esperado del factor λ (véase Ecuación 15) y ajusta el operador de mutación para seguir la curva de crecimiento teórica y alcanzar el *takeover time* en el instante $t_{max} = t_c$.

Los resultados muestran el rendimiento mejorado de este nuevo algoritmo con respecto al método basado en memoria global, denominado $reEA$ (véase 0). Para un periodo de cambio bajo ($\tau = 50$), el $reEA_{\mu_m}$ encuentra mejores soluciones en el espacio de búsqueda, incluso cuando se experimentan cambios severos (esto se puede ver para $\rho = 0.8$ en la 0). Para períodos de cambio grandes, el $reEA_{\mu_m}$ es capaz de encontrar el óptimo global, y seguirlo a lo largo de varios intervalos estacionarios. Estos resultados se deben a un mejor equilibrio entre exploración y explotación promovido por el nuevo algoritmo, el cual es ajustado automáticamente a través de los modelos de convergencia teóricos en relación con las características del problema dinámico.

Como conclusión, se validó que construir modelos teóricos de takeover es una cuestión importante en EDO, tanto por el propio modelo como por las líneas de investigación y aplicación que abre en dirección de la construcción de nuevos algoritmos con un fundamento matemático.

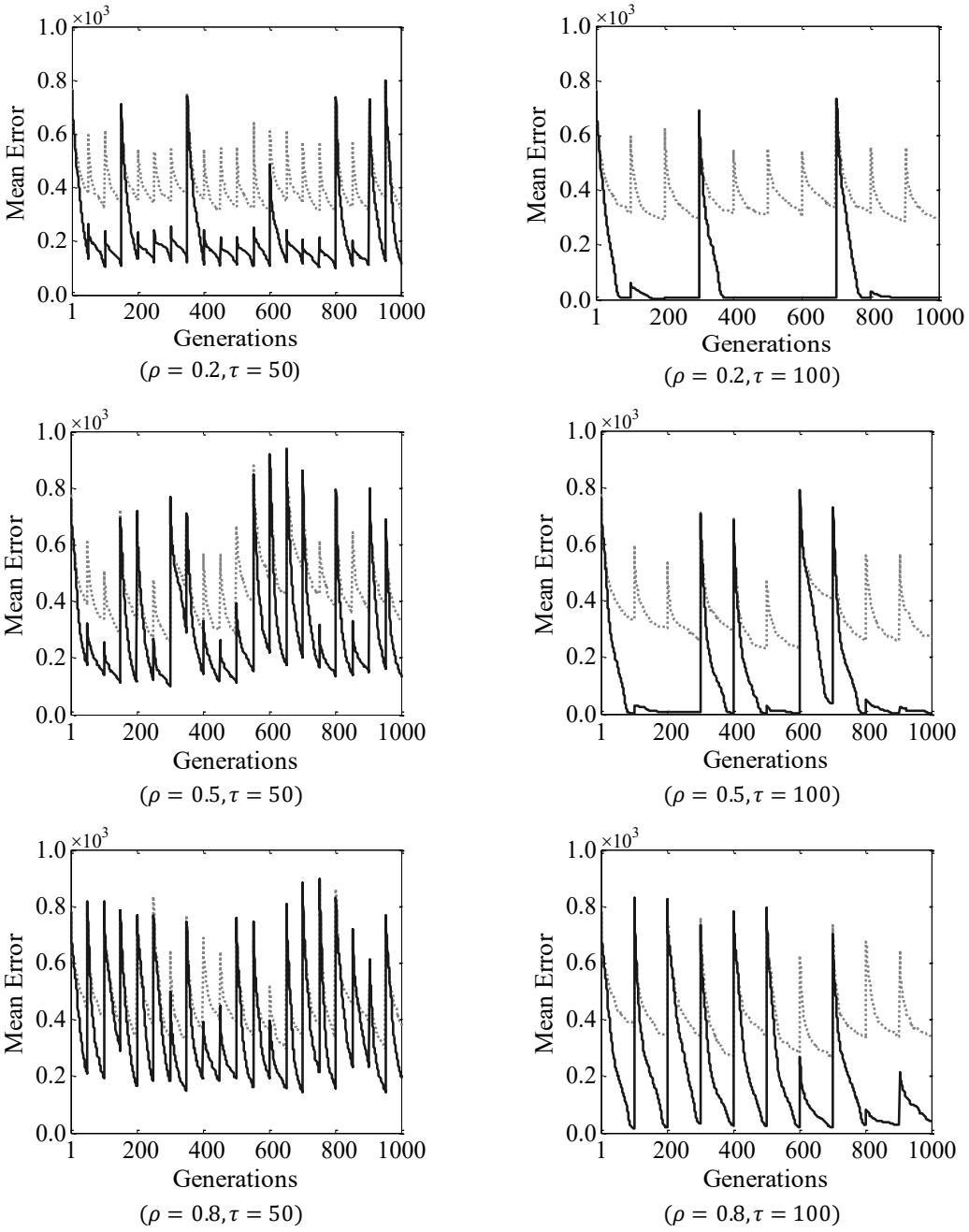


Figura 14. Rendimiento mejorado del nuevo algoritmo adaptativo $reEA_{\mu_m}$ (curva continua) con respecto al algoritmo sin el mecanismo de adaptación $reEA$ (curva discontinua).

3.6 Movilidad inteligente por la optimización del tiempo en rojo de los semáforos

Y. Bravo, J. Ferrer, G. Luque, E. Alba
Smart Cities, 2016
DOI: 10.1007/978-3-319-39595-1_15

Finalmente, tras haber publicado resultados metodológicos (en las publicaciones 1 y 2) y de fundamentos (en las publicaciones 3 y 4), abordamos la aplicación de técnicas evolutivas a problemas del mundo real. De esta forma comenzamos a abordar la RQ 6 aunque no directamente aplicando técnicas de optimización dinámica, siendo esta la contribución principal de otra publicación en curso también dentro del ámbito de la Movilidad Inteligente.

En la publicación que ahora hacemos referencia presentamos un sistema de apoyo a la toma de decisión denominado *HITUL*, siglas del nombre en inglés Holistic Intelligence for Traffic Urban Lights, que ayuda a los gestores del tráfico a generar planes semafóricos óptimos para las ciudades reales. Esto es, el sistema realiza la optimización de los tiempos en rojo de los semáforos en toda la ciudad. Lo cual tiene una importancia considerable en términos de consumo de energía, gestión del flujo de tráfico, seguridad de los peatones y cuestiones ambientales. La Figura 15 muestra una perspectiva funcional de HITUL.

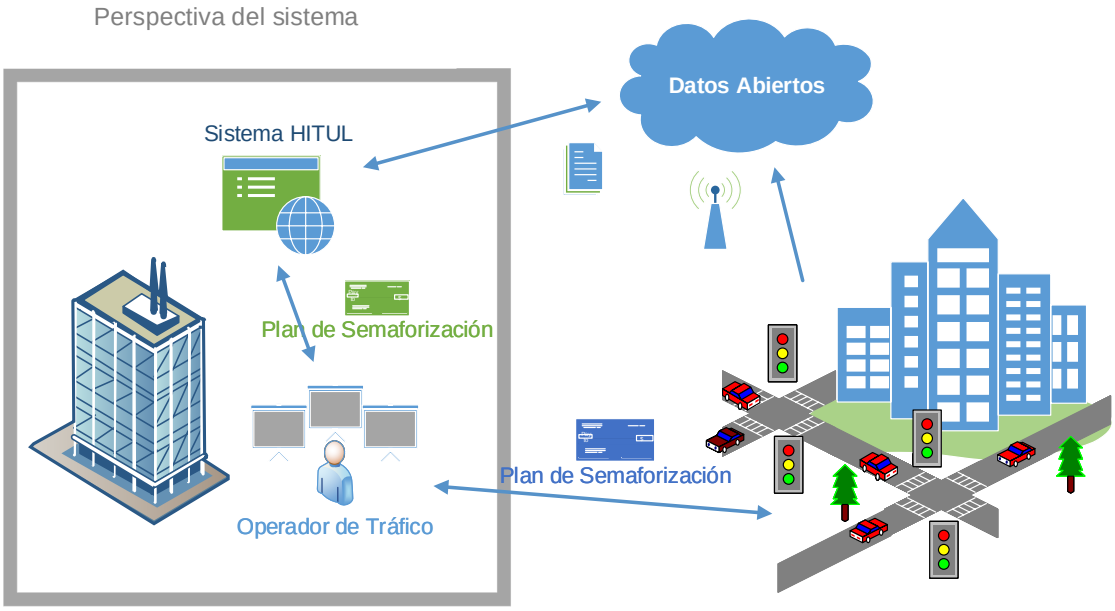


Figura 15. Perspectiva del sistema HITUL.

El sistema propuesto aborda la mayoría de los inconvenientes de los actuales sistemas existentes considerando zonas muy grandes e información realista. Además, el sistema proporciona facilidades para la toma de decisiones de los gestores de tráfico, como permitir la comparación entre planes de tráfico, resaltar la importancia de cada semáforo en el plan final, o permitir restringir la zona a considerar, el perfil de tráfico utilizado o incluso el objetivo principal que quiere optimizar. La validación del sistema se hizo en Málaga, una ciudad de tamaño medio, donde se demostró que HITUL es capaz de brindar un verdadero apoyo a los responsables de la toma de decisiones en la planificación de la red de semáforos de la ciudad.

Esta publicación está más relacionada con el trabajo futuro de esta tesis doctoral, donde se pretende abordar este problema como caso de problemas dinámicos real. De hecho, la configuración óptima de los semáforos depende del flujo vehicular de la ciudad, y este cambia con el tiempo. Luego, la optimización de los tiempos en rojo de los semáforos de la ciudad es un problema de optimización dinámico de mucho interés.

En un primer acercamiento consideramos perfiles del tráfico previamente establecidos, que nos brindan las intensidades vehiculares de la ciudad. Con el desarrollo de aplicaciones de sensorización que se está llevando a cabo en nuestro grupo de investigación, será posible detectar el flujo en tiempo real. Es al momento de enfrentar el problema dinámico donde planeamos usar los nuevos algoritmos diseñados en publicaciones anteriores.

Figura 16. Diferencias entre el plan base y el plan optimizado por el sistema HITUL en toda Málaga (961 intersecciones). El color *rojo* indica una gran diferencia, *amarillo* una diferencia media y *verde* poca diferencia.

La Figura 16 muestra el resultado de la optimización de los semáforos en toda la ciudad de Málaga. En esta se puede observar las diferencias con respecto a un plan base que sigue criterios de expertos. Los resultados confirman que un enfoque de optimización holística a lo largo de toda la ciudad es mejor que políticas de control locales a fin de mejorar el tráfico vehicular, eliminando atascos, contaminación, ruido, etc.

3.7 Movilidad inteligente mediante el uso del vehículo compartido en tiempo real (*Real-time Ridesharing*)

Y. Bravo, G. Luque, E. Alba

Trabajo en curso

También con propósito de aplicar los resultados de nuestros estudios metodológicos y teóricos en aplicaciones reales en el área de la Movilidad Inteligente (RQ 6) iniciamos la investigación que resumimos en esta sección. En este caso sí tratamos el problema como dinámico y hacemos una interpretación de algunos fundamentos desarrollados y que tienen una aplicación directa en escenarios reales, específicamente de *takeover time* (véase Sección 3.4 y 3.5).

El primer paso para aplicar los nuevos algoritmos desarrollados fue implementar una plataforma de optimización específica para en DOP reales de transporte y movilidad inteligente. Esta plataforma, implementada en lenguaje C++11, provee estructuras de datos y algoritmos que facilitan la representación y manejo de soluciones sobre mapas de carreteras. Inicialmente está orientada a Algoritmos Evolutivos, pero permite (por el uso de la programación genérica) parametrizar los algoritmos y componentes principales para crear nuevos algoritmos. Finalmente, incluye un simulador de eventos discretos para notificar al algoritmo los cambios del problema en la medida que estos ocurren.

El primer DOP real que abordamos es un servicio para compartir vehículos en tiempo real (con poco tiempo de antelación), conocido por su término en inglés como *real-time ridesharing* o *carpooling* dinámico (d'Orey, Fernandes, and Ferreira 2012). Estos sistemas promueven la reducción del número de vehículos que transitan por la ciudad. De esta forma, permiten ahorrar combustible, disminuir los atascos y en consecuencia reducir las emisiones de CO2. Concretamente, abordamos un caso práctico del uso compartido del Taxi en una ciudad de gran tamaño como Beijing, cuya red vehicular contiene 106.579 intersecciones y 141.380 segmentos de carretera (Ma, Zheng, and Wolfson 2013). Los datos están generados a partir de las trayectorias de más de 33.000 taxis durante un periodo de 87 días entre marzo y mayo de 2011. La Figura 17 muestra el número promedio de llamadas de Taxi y el tiempo promedio entre llamadas a lo largo de un día.

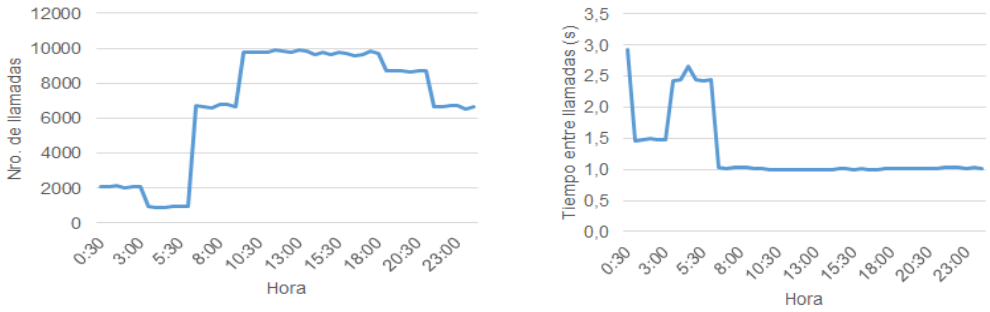


Figura 17. Número promedio de llamadas de taxi y tiempo entre llamadas en un día en la ciudad de Beijing (Ma, Zheng, and Wolfson 2013).

El número de llamadas de Taxi que se recibe varía a lo largo del día (véase Figura 17, izquierda), lo que sugiere que la técnica usada para resolver la asignación de personas a vehículos compartidos debería adaptarse a condiciones variables. En este estudio también se analiza el tiempo entre llamadas (véase Figura 17, derecha), corroborando el posible beneficio de ajustar el tiempo de ejecución del algoritmo a la frecuencia de llamadas en tiempo real.

Los resultados de fundamentos relacionados con la definición de *takeover time* in EDO son aplicable en estos casos, donde se tiene alguna información sobre la frecuencia con que cambia el problema (véase Sección 3.5). Específicamente, es posible crear un algoritmo evolutivo reactivo adaptativo que busque una planificación óptima con los recursos actuales, antes de incorporar en el sistema las nuevas solicitudes de carreras. Lo cual como se demostró anteriormente en problemas de prueba, puede aumentar el rendimiento del algoritmo.

No obstante, puede ocurrir que los cambios ocurran con una frecuencia muy elevada, como ocurre en el problema de referencia. La Figura 16 (derecha) indica que la mayor variación de tiempo entre dos solicitudes consecutivas es de 3 segundos aproximadamente (muy pequeña), lo que es específico para grandes ciudades como Beijing. Cuanto mayor sea la ciudad hay más probabilidad de recibir llamadas de Taxi simultáneas desde diferentes ubicaciones geográficas. En estos casos puede que el algoritmo no tenga tiempo suficiente para evolucionar y optimizar la planificación con los recursos actuales, antes de incorporar las nuevas solicitudes.

Una de las soluciones que estamos estudiando para resolver esta limitación es la acumulación de eventos en un tiempo razonable, en función del *takeover time* esperado. De esta forma, los nuevos pedidos no se insertan en la planificación en el momento que se reciben (que puede ser por cada segundo), sino que se coleccionan para insertarse en al alcanzarse el *takeover time* esperado. Aquí debemos tener en cuenta un tiempo máximo de servicio, que está determinado por el tipo de problema. Para una solicitud de Taxi, consideramos que la aplicación no debe esperar más de 15 segundos para realizar una asignación.

En conclusión, esta investigación aún en curso ya va corroborando la importancia de aplicar las técnicas EDO que diseñamos a problemas dinámicos reales de la ciudad. La razón es que pueden aparecer características y condiciones no tomadas en cuenta en los estudios metodológicos o teóricos que realizamos basados en plataformas de prueba.

4 CONCLUSIONES Y TRABAJO FUTURO

Los problemas dinámicos requieren estudios más allá del actual buen rendimiento de algoritmos eficientes para resolver una instancia estática del problema a la vez, ya que son en general incapaces de adaptarse a los cambios continuos del problema. De hecho, los problemas dinámicos están fuera del alcance del algoritmo evolutivo estándar debido al *problema de la convergencia* (Yang 2008), en la medida en que convergen a un único punto del espacio de búsqueda, ellos pierden la diversidad necesaria para reaccionar y adaptarse después de un cambio en el problema. A pesar de no estar directamente enfocados en problemas dinámicos, los algoritmos evolutivos también ofrecen muchas ventajas (simplicidad, robustez, paralelismo natural, etc.) que se pueden aprovechar en optimización dinámica si se diseñan métodos eficientes y eficaces para eliminar tales limitaciones. Esta es la motivación principal de los investigadores en el campo de la optimización dinámica evolutiva y de la presente tesis doctoral.

Esta investigación abordó el diseño, implementación y evaluación de nuevos algoritmos evolutivos para problemas de optimización dinámicos. Uno de estos algoritmos que demostró ser muy beneficioso para la optimización dinámica fue el algoritmo paralelo distribuido (dEA), también conocidos como modelo de islas. Dicho modelo descentraliza la población en componentes que evolucionan en paralelo e intercambian información mediante *migraciones*. Específicamente, nosotros mostramos cómo se pueden ajustar dos parámetros clave de la política de migración para diseñar mejores algoritmos en optimización dinámica, que han sido capaces de enfrentar distintos comportamientos dinámicos con éxito.

Una cuestión importante en el diseño del dEA es la decisión sobre la frecuencia para migrar los individuos. Nuestro trabajo en esta dirección demostró que éste es un parámetro clave en el rendimiento del algoritmo en optimización dinámica, que afecta no solo al rendimiento del algoritmo para encontrar una solución, sino que influye directamente en la capacidad para adaptarse a cambios continuos en un problema dinámico. Por ejemplo, migrar en cada iteración del algoritmo (total acoplamiento entre las islas) mejora la búsqueda de algoritmo secuencial, debido a la explotación en paralelo de todas las islas. Sin embargo,

esta configuración no es buena para todas las instancias de DOP, porque asemeja el comportamiento *panmíctico* de una población centralizada y no escapa al problema de la convergencia prematura. Como conclusión, frente a DOP con alta severidad de cambio o multimodales es mejor usar un periodo de migración alto. Esto garantiza la reproducción separada de las islas y mejora la diversidad global y la especiación, dos características que hemos probado aquí son muy favorables para EDO. En particular, resulta interesante la ventaja de migrar cuando se detecta un cambio en el problema, como mecanismo para reaccionar y adaptarse a dicho cambio.

Otro parámetro importante en el diseño e implementación del dEA es la forma en que se seleccionan y se reemplazan las soluciones que migran entre las islas. Por un lado, los resultados promueven tanto la selección como el reemplazo de migrantes, en base a los individuos menos aptos de cada subpoblación. El rendimiento fue notablemente mejor al resolver DOP unimodales (un solo óptimo) con pequeños cambios (baja severidad de cambio). Sin embargo, esta política de migración muestra una pérdida de diversidad global, lo que denota dificultad para perseguir los movimientos del óptimo durante mucho tiempo. Por otro lado, la estrategia de reemplazo basada en distancia (el más alejado de cada subpoblación) mostró ser más robusta para rastrear un mayor número de óptimos locales, mejorando la especiación de la población. Sin embargo, no es bueno para explorar un espacio de búsqueda dinámico, ya que produce la pérdida temprana de la diversidad local.

Por lo tanto, hemos visto cómo diferentes políticas de migración pueden ayudar a diseñar nuevos dEA para lograr un equilibrio entre la convergencia (adaptarse rápidamente a los movimientos de los óptimos) y la especiación (búsqueda de múltiple subóptimos al mismo tiempo). Con ello confirma la importancia de perseguir múltiples valores subóptimos en la optimización dinámica, lo que corresponde a nuestro interés en el modelo paralelo para DOP. Además, los resultados presentados en estos estudios cierran algunas brechas en la comprensión del comportamiento de dEA y, en general, todos los enfoques multi-poblacionales para DOP que intercambian algún tipo de información entre las subpoblaciones. Al mismo tiempo, esto abre nuevas líneas de investigación en el desarrollo de enfoques adaptativos o auto-adaptativos multi-poblacionales en EDO, capaces de manejar diferentes comportamientos dinámicos, lo cual es otro desafío actual en el campo.

También desde el desafío metodológico, estudiamos otro de los métodos muy utilizados para enfrentarse a DOP, los métodos que usan *memoria global*, con el fin de almacenar soluciones que luego son usadas para perseguir el óptimo. Dos aspectos claves en el diseño de estos métodos son el reemplazo y la recuperación de las soluciones de la memoria. Nuestro estudio, basado en un número

representativo de criterios, de forma independiente y combinados también, descubrió cuál es el efecto de cada estrategia en el rendimiento del EA en DOP. Cuando se probaron varias estrategias combinadas (muy común en esquemas de memoria reciente) se demostró que el orden en que se combinan las estrategias de memoria es importante. Especialmente la contribución a la diversidad de la memoria resultó ser una prioridad en el diseño de algoritmo, en contraposición con algunas implementaciones existentes, como la presentada en (Lepagnot et al. 2013). Estas implementaciones pudieran ahora beneficiarse de nuestro estudio para diseñar mejores algoritmos basados en memoria.

En esta tesis doctoral también desarrollamos fundamentos teóricos, enfrentando otro de los desafíos actuales en el campo de la computación dinámica evolutiva. Nosotros definimos y medimos de forma efectiva la presión selectiva de los algoritmos evolutivos en DOP. Para ello, redefinimos el concepto de *takeover time*, una métrica muy usada para medir presión selectiva en optimización estática, que no tenía una noción clara EDO. Esto es porque el *takeover* depende directamente de la mejor solución de la población y, en un problema dinámico, la "mejor solución" cambia continuamente. Como otra de nuestras contribuciones, y dando respuesta a preguntas de investigación planteadas, introducimos una definición clara para takeover time en problemas dinámicos. Esta se basa en el valor esperado de takeover en cada periodo estacionario.

Construir modelos de takeover demostró ser otra cuestión importante en EDO, tanto por el modelo en sí que nos permite conocer mejor el funcionamiento de las técnicas como porque abre nuevas líneas de investigación en la construcción de enfoques algorítmicos con una base matemática. Este ha sido el foco de muchos estudios teóricos en la optimización estática; Sin embargo, todavía era un asunto abierto en EDO. Un problema importante es que la mejor clase de fitness puede cambiar con el tiempo, debido a los cambios continuos del problema. Por lo tanto, además de la dinámica de la población, los modelos de takeover time deben considerar la dinámica de cambio.

La definición propuesta de *takeover time* es clara y efectiva para caracterizar el comportamiento de los algoritmos evolutivos en DOP. Además, no solo definimos el concepto de *takeover time*, también desarrollamos modelos matemáticos para calcular su valor, siendo esto otra de las cuestiones de investigación que nos planteamos. Nuestros modelos tienen en cuenta tanto características del EA como del DOP. Lo cual tiene mucha aplicación en EDO, como herramienta para conocer mejor los métodos actuales y cómo influyen sus parámetros y los del problema en el rendimiento de estos. Así mismo, estos modelos permiten diseñar nuevos métodos adaptativos, capaces de ajustar el comportamiento del algoritmo evolutivo a la dinámica de cambio.

De hecho, en nuestra investigación no solo proponemos modelos de takeover para DOP, sino que también los usamos para construir un nuevo EA para EDO, denominado $meEA_{\mu_m}$. Este algoritmo se basa en los modelos de takeover para ajustar el operador de mutación y así controlar la convergencia del del EA a lo largo de la ejecución, adaptándose a distintos periodos y severidades de cambio en un DOP. Los resultados han validado las ventajas del nuevo algoritmo sobre el método tradicional de generar diversidad después de un cambio.

Finalmente, nos planteamos aplicar los resultados metodológicos y de fundamentos en el diseño de nuevos algoritmos para optimización dinámica con aplicación a problemas reales. De hecho, este no es un objetivo independiente en nuestra investigación, el enfoque de cada estudio en la futura aplicación a problemas reales ha estado presente desde el inicio. Esto se evidencia en los esfuerzos por diseñar, implementar y evaluar algoritmos capaces de adaptarse de a distintos tipos de problemas dinámicos.

También en el marco de esta tesis hemos desarrollado el sistema HITUL, para la optimización de los semáforos de una ciudad. Este resultado se relaciona con el trabajo futuro, que en gran medida ya se ha iniciado para seguir trabajando en las líneas de impacto claras de aplicación real que se derivan de esta tesis. Un desarrollo en esta dirección aún no terminado en el momento de presentar la tesis, es una aplicación para el uso del vehículo compartido o *carpooling*. Este sistema facilita compartir un automóvil con otras personas que viajan en la misma dirección, y ayudan así a reducir los atascos y las emisiones de CO_2 al reducir el número de coches en la red vial. La aplicación que estamos desarrollando asigna personas y vehículos de forma óptima, en un contexto inherentemente dinámico de la demanda de pasajeros. En esta investigación en curso se están empleando los dEA y las técnicas basadas en *takeover time* desarrollada en esta tesis.

En resumen, varias direcciones de trabajo futuro se conciben a partir de la presente investigación. En relación al diseño de modelos paralelos distribuidos en EDO, hay otros parámetros no examinados en esta tesis el dEA, tales como la homogeneidad o la sincronización. Estos parámetros pudieran ser abordados en un futuro para ganar mayor comprensión sobre el rendimiento del dEA en EDO, y promover su uso en el dominio.

Por otro lado, basado en las correlaciones de los parámetros del algoritmo y las características de los DOP aquí estudiadas, nos planteamos como trabajo futuro crear un marco común que sirva para la caracterización en tiempo real de los problemas dinámicos, a partir del rendimiento de uno o más algoritmos que lo resuelven. Esto también aplica a los nuevos algoritmos sugeridos basados en memoria global. Como resultado, pretendemos diseñar e implementar técnicas cada vez más eficientes en DOP hábiles para enfrentar las complejidades de las aplicaciones a problemas reales.

Finalmente, hemos trabajado durante el tiempo de tesis en dotar a la investigación y a los resultados software de un realismo y usabilidad reales. Esto no es visible en publicaciones aún, pero nos está permitiendo desarrollar una aplicación para posible transferencia al mercado industrial en el dominio de la movilidad inteligente en entornos urbanos. Para ello, hemos realizado una confección de mapas con datos reales tomados de varias fuentes (*open data*), una simulación orientada a evaluar políticas y decisiones de movilidad (una especie de función de *fitness* avanzada muy compleja) y hemos incluido los algoritmos desarrollados en esta tesis. Este trabajo futuro es ya una realidad hoy en día al momento de la defensa, arrojando resultados de mucho interés. Esperamos en poco tiempo tras la defensa de esta tesis doctoral estar en condiciones de explotar esto, ya que algunas empresas están interesadas en tal producto, permitiendo una transferencia universidad-empresa y teoría-práctica ideal como punto final a todo este trabajo.

ANEXO A – Relación de méritos y publicaciones que avalan el trabajo de tesis

En este anexo presentamos el conjunto de artículos científicos que se han publicado durante los años en el que se ha desarrollado esta tesis. Estas publicaciones hablan por el interés, validez e impacto en la comunidad científica. Han aparecido en foros de impacto y han sido sometidos a revisión por expertos por parte de investigadores expertos.

En total son 6 publicaciones científicas: 2 en revistas indexadas y 4 en congresos.

A.1 Publicaciones en revistas de JCR

- Y. Bravo, G. Luque, E. Alba (2015). **"Takeover Time in Evolutionary Dynamic Optimization: from Theory to Practice"**. En Applied Mathematics and Computation (AMC), Volumen 250, Número 1, pp. 94-104
  - ➢ DOI: 10.1016/j.amc.2014.10.107
  - ➢ Factor de Impacto: 1,1345
  - ➢ Categoría: APPLIED MATHEMATICS: 54/254 (Q1)

- Y. Bravo, G. Luque, E. Alba (2016). **"Global Memory Schemes for Dynamic Optimization"**. En Natural Computing, Volumen 15 Número 2, pp. 319-333
  - ➢ DOI: 10.1007/s11047-015-9497-2
  - ➢ Factor de Impacto: 1,310
  - ➢ Categoría: COMPUTER SCIENCE, THEORY & METHODS: 40/105 (Q2)

A.2 Publicaciones en congresos

- Y. Bravo, G. Luque, E. Alba (2012). **"Influence of the Migration Period in Parallel Distributed GAs for Dynamic Optimization"**. En Learning and Intelligent Optimization (LION), 6th International Conference, LNCS 7219, Paris, France, Springer Berlin Heidelberg, pp. 343-348.

  - DOI: 10.1007/978-3-642-34413-8_25

  - Breve historial del congreso: LION tiene una madurez de diez ediciones, con ratio de aceptación es del 34%. Entre los miembros del comité técnico se encuentran investigadores muy reconocidos en el campo de la Optimización: Pascal Van Hentenryck, Marc Schoenauer, Zbigniew Michalewicz, Francisco Herrera, Carlos Coello Coello.

- Y. Bravo, G. Luque, E. Alba (2013). **"Migrants Selection and Replacement in Distributed Evolutionary Algorithms for Dynamic Optimization"**. En Distributed Computing and Artificial Intelligence (DCAI), 10th International Conference, Advances in Intelligent Systems and Computing 217, Springer International Publishing, pp. 155-162.

  - DOI: 10.1007/978-3-319-00551-5_19

  - Editores de las actas: Sigeru Omatu, José Neves, Juan M. Corchado Rodriguez, Juan F Paz Santana y Sara Rodríguez Gonzalez

- Y. Bravo, G. Luque, E. Alba (2013). **"Takeover Time in Dynamic Optimization Problems"**. En Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2013 IEEE Symposium, Computational Intelligence (SSCI) 2013, Singapore, IEEE, pp 25-30.

  - DOI: 10.1109/CIDUE.2013.6595768

  - IEEE CIDUE 2013 se realiza en el marco de IEEE SSCI 2013, uno de los dos eventos internacionales bienales patrocinados por la IEEE Computational Intelligence Society (IEEE CIS), que promueve todos los aspectos de la teoría y las aplicaciones de la inteligencia computacional. En particular, la sesión de CIDUE se centra en el objeto de estudio de la tesis: Problemas de Optimización Dinámicos. Entre los editores se encuentran Yoachu Jin y Shengxiang Yang, pioneros en este campo.

  - Editores de las actas: Yaochu Jin, Robi Polikar, Shengxiang Yang

- Y. Bravo, J. Ferrer, G. Luque, E. Alba (2016). **"Smart Mobility by Optimizing the Traffic Lights: A New Tool for Traffic Control Centers"**. En Smart Cities, LNCS 9704 of the series Lecture Notes in Computer Science, Málaga, España, Springer International Publishing, pp 147-156.
    - DOI: 10.1007/978-3-319-39595-1_15
    - Es el primer Smart-CT que se celebra.
    - Editores de las actas: Enrique Alba, Francisco Chicano y Gabriel Luque

REFERENCIAS

Alba, Enrique, Christian Blum, Pedro Isasi, Coromoto Len, and Juan Antonio Gmez. 2009. *Optimization Techniques for Solving Complex Problems*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Alba, Enrique, Amir Nakib, and Patrick Siarry. 2013. *Metaheuristics for Dynamic Optimization*. Edited by Enrique Alba, Amir Nakib, and Patrick Siarry. Vol. 433. Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg.

Alba, Enrique, and José M. Troya. 2000. "Influence of the Migration Policy in Parallel Distributed GAs with Structured and Panmictic Populations." *Applied Intelligence* 12 (3). Springer Netherlands: 163–81.

Arnold, Dirk V., and Hans-Georg Beyer. 2002. "Random Dynamics Optimum Tracking with Evolution Strategies." In *Parallel Problem Solving from Nature — PPSN VII*, 3–12. Springer Berlin Heidelberg.

Ayvaz, Demet, Haluk Topcuoglu, and Fikret Gurgen. 2006. "Hybrid Techniques for Dynamic Optimization Problems." In *Computer and Information Sciences - ISCIS 2006*, edited by Albert Levi, Erkay Savas, Hüsnü Yenigün, Selim Balcisoy, and Yücel Saygin, 4263:95–104. Lecture Notes in Computer Science. Springer Berlin / Heidelberg.

Bäck, Thomas, David B. Fogel, and Zbigniew Michalewicz. 1997. *Handbook of Evolutionary Computation*. Adam Hilger, IOP Publishing.

Ben-Romdhane, Hajer, Enrique Alba, and Saoussen Krichen. 2013. "Best Practices in Measuring Algorithm Performance for Dynamic Optimization Problems." *Soft Computing* 17 (6): 1005–17.

Bosman, Peter A. N. 2005. "Learning, Anticipation and Time-Deception in Evolutionary Online Dynamic Optimization." In *Proceeding GECCO '05 Proceedings of the 7th Annual Workshop on Genetic and Evolutionary Computation*, 39–47.

———. 2007. "Learning and Anticipation in Online Dynamic Optimization." In *Evolutionary Computation in Dynamic and Uncertain Environments*, 51:129–52. Springer Berlin Heidelberg.

Branke, Jürgen. 1999. "Memory Enhanced Evolutionary Algorithms for Changing Optimization Problems." *Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999* 3. IEEE: 1875–82.

———. 2001. "Evolutionary Approaches to Dynamic Optimization Problems-Updated Survey." *....Evolutionary Algorithms for Dynamic Optimization*, 2–6.

Branke, Jürgen, Thomas Kaussler, Christian Schmidt, and Hartmut Schmeck. 2000. "A Multi-Population Approach to Dynamic Optimization Problems." In *In Adaptive Computing in Design and Manufacturing*, 299–308. Springer.

Branke, Jürgen, and Hartmut Schmeck. 2003. “Designing Evolutionary Algorithms for Dynamic Optimization Problems.” University of Karlsruhe: Institute AIFB. doi:10.1007/978-3-642-18965-4_9.
Branke, Jürgen, and Wei Wang. 2003. “Theoretical Analysis of Simple Evolution Strategies in Quickly Changing Environments.” In *Genetic and Evolutionary Computation — GECCO 2003*, 537–48. Springer Berlin Heidelberg.
Bravo, Yesnier, Gabriel Luque, and Enrique Alba. 2016. “Global Memory Schemes for Dynamic Optimization.” *Natural Computing* 15 (2): 319–33.
Brest, Janez, Ales Zamuda, Borko Boskovic, Mirjam Sepesy Maucec, and Viljem Zumer. 2009. “Dynamic Optimization Using Self-Adaptive Differential Evolution.” In *2009 IEEE Congress on Evolutionary Computation*, 415–22. IEEE.
Cantú-paz, Erick. 1999. “Migration Policies and Takeover Times in Parallel Genetic Algorithms.” http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.45.8075.
Cheng, Hui, and Shengxiang Yang. 2010. “Multi-Population Genetic Algorithms with Immigrants Scheme for Dynamic Shortest Path Routing Problems in Mobile Ad Hoc Networks.” In *Evo*, 562–71. Springer Berlin Heidelberg.
Cheng, Hui, Shengxiang Yang, and Xingwei Wang. 2012. “Immigrants-Enhanced Multi-Population Genetic Algorithms for Dynamic Shortest Path Routing Problems in Mobile Ad Hoc Networks.” *Applied Artificial Intelligence: AAI* 26 (7): 673–95.
Cobb, Helen G. 1990. “An Investigation into the Use of Hypermutation as an Adaptive Operator in Genetic Algorithms Having Continuous, Time-Dependent Nonstationary Environments.” In *Navy Center for Applied Research in Artificial Intelligence*, 19.
Cruz, Carlos, Juan R. González, and David A. Pelta. 2011. “Optimization in Dynamic Environments: A Survey on Problems, Methods and Measures.” *Soft Computing* 15 (7): 1427–48.
De Jong, Kenneth. 1999. “Evolving in a Changing World.” In *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1609:512–19. London, UK: Springer-Verlag.
Dorigo, Marco, Vittorio Maniezzo, and Alberto Colorni. 1996. “Ant System: Optimization by a Colony of Cooperating Agents.” *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics: A Publication of the IEEE Systems, Man, and Cybernetics Society* 26 (1): 29–41.
Droste, Stefan. 2004. “Analysis of the (1 + 1) EA for a Noisy OneMax.” In *Genetic and Evolutionary Computation – GECCO 2004 SE - 107*, 3102:1088–99. IEEE.
Eggermont, Jeroen, Tom Lenaerts, Sanna Poyhonen, and Alexandre Termier. 2001. “Raising the Dead: Extending Evolutionary Algorithms with a Case-Based Memory.” In *Genetic Programing*, 280–90.
Fogel, David B. 1997. “The Advantages of Evolutionary Computation.” In *Biocomputing and Emergent Computation: Proceedings of BCEC97*, 1–11. World Scientific Press.
García, Salvador, Daniel Molina, Manuel Lozano, and Francisco Herrera. 2008. “A Study on the Use of Non-Parametric Tests for Analyzing the Evolutionary Algorithms’ Behaviour: A Case Study on the CEC’2005 Special Session on

Real Parameter Optimization." *Journal of Heuristics* 15 (6): 617–44.
Glover, Fred, and Gary A. Kochenberger. 2003. *Handbook of Metaheuristics*. Vol. 57. International Series in Operations Research & Management Science. Boston: Kluwer Academic Publishers.
Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Vol. Addison-We.
———. 2002. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers Norwell.
Goldberg, David E., and Kalyanmoy Deb. 1991. "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms." In *Foundations of Genetic Algorithms*, 69–93.
Grefenstette, John J. 1992. "Genetic Algorithms for Changing Environments." *Parallel Problem Solving from Nature 2* 2 (1992): 137–44.
Halder, U., S. Das, and D. Maity. 2013. "A Cluster-Based Differential Evolution Algorithm With External Archive for Optimization in Dynamic Environments." *IEEE Transactions on Cybernetics* 43 (3): 881–97.
Haobo, Fu, P. R. Lewis, B. Sendhoff, Tang Ke, and Yao Xin. 2014. "What Are Dynamic Optimization Problems?" In *Congress on Evolutionary Computation (CEC), 2014*, 1550–57.
Homayounfar, H., S. Areibi, and F. Wang. 2003. "An Advanced Island Based GA For Optimization Problems." In *Proceedings of the International DCDIS Conference on Engineering Applications and Computations*, 46–51.
Jin, Yaochu. 2004. *Evolutionary Computation in Dynamic and Uncertain Environments*. Edited by Shengxiang Yang; Yew-Soon Ong; Yoachu Jin. Vol. 51. Springer.
———. 2005. "A Comprehensive Survey of Fitness Approximation in Evolutionary Computation." *Soft Computing* 9 (1): 3–12.
Jin, Yaochu, and Jürgen Branke. 2005. "Evolutionary Optimization in Uncertain Environments—A Survey." *IEEE Transactions on Evolutionary Computation* 9 (3): 303–17.
Kennedy, J., and R. Eberhart. 1995. "Particle Swarm Optimization." *Neural Networks, 1995. Proceedings., IEEE International Conference on* 4: 1942–48 vol.4.
Lepagnot, Julien, Amir Nakib, Hamouche Oulhadj, and Patrick Siarry. 2013. "A Multiple Local Search Algorithm for Continuous Dynamic Optimization." *Journal of Heuristics* 19 (1): 35–76.
Li, Changhe, and Shengxiang Yang. 2008a. "A Generalized Approach to Construct Benchmark Problems for Dynamic Optimization." In *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5361 LNAI:391–400.
———. 2008b. "An Island Based Hybrid Evolutionary Algorithm for Optimization." In *Proceeding SEAL '08 Proceedings of the 7th International Conference on Simulated Evolution and Learning*, 180–89. Berlin, Heidelberg: Springer Berlin Heidelberg.
Li-Ning Xing, P. Rohlfshagen, Ying-Wu Chen, and Xin Yao. 2011. "A Hybrid Ant Colony Optimization Algorithm for the Extended Capacitated Arc Routing Problem." *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics: A Publication of the IEEE Systems, Man, and Cybernetics Society* 41 (4): 1110–23.

Lopes, Raul H. C. 2011. “Kolmogorov-Smirnov Test.” In *International Encyclopedia of Statistical Science*, 718–20.
Luque, Gabriel, and Enrique Alba. 2011. *Parallel Genetic Algorithms*. Vol. 367. Studies in Computational Intelligence. Berlin, Heidelberg: Springer.
———. 2013. “Math Oracles.” In *Proceeding of the Fifteenth Annual Conference Companion on Genetic and Evolutionary Computation Conference Companion - GECCO ’13 Companion*, 217. New York, New York, USA: ACM Press.
Ma, Shuo, Yu Zheng, and Ouri Wolfson. 2013. “T-Share: A Large-Scale Dynamic Taxi Ridesharing Service.” In *Proceedings - International Conference on Data Engineering*. doi:10.1109/ICDE.2013.6544843.
Mavrovouniotis, Michalis, and Shengxiang Yang. 2012. “Ant Colony Optimization with Memory-Based Immigrants for the Dynamic Vehicle Routing Problem.” In *2012 IEEE Congress on Evolutionary Computation*, 1–8. IEEE.
Michalewicz, Zbigniew. 1996. *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin, Heidelberg: Springer Berlin Heidelberg.
Miller, Brad L., and David E. Goldberg. 1995. “Genetic Algorithms, Tournament Selection, and the Effects of Noise.” *Complex Systems* 9 (3): 193–212.
Morrison, Ronald W. 2004. *Designing Evolutionary Algorithms for Dynamic Environments*. Springer.
Nguyen, Trung Thanh, Shengxiang Yang, and Jürgen Branke. 2012. “Evolutionary Dynamic Optimization: A Survey of the State of the Art.” *Swarm and Evolutionary Computation* 6. Elsevier: 1–24.
Nguyen, Trung Thanh, Shengxiang Yang, Jürgen Branke, and Xin Yao. 2013. “Evolutionary Dynamic Optimization: Methodologies.” *Studies in Computational Intelligence* 490: 39–64.
Nguyen, Trung Thanh, and Xin Yao. 2009. “Dynamic Time-Linkage Problems Revisited.” *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. doi:10.1007/978-3-642-01129-0_83.
Okabe, Tatsuya. 2007. “Theoretical Analysis of Selection Operator in Genetic Algorithms.” In *2007 IEEE Congress on Evolutionary Computation*, 4676–83.
Oliveto, Pietro S., and Christine Zarges. 2015. “Analysis of Diversity Mechanisms for Optimisation in Dynamic Environments with Low Frequencies of Change.” *Theoretical Computer Science* 561 (PA): 37–56.
Oppacher, Franz, and Mark Wineberg. 1999. “The Shifting Balance Genetic Algorithm: Improving the GA in a Dynamic Environment.” In *Proceedings of the Genetic and Evolutionary Computation Conference*, 1:504–10. San Francisco: Morgan Kaufman.
Orey, Pedro M. d’, Ricardo Fernandes, and Michel Ferreira. 2012. “Empirical Evaluation of a Dynamic and Distributed Taxi-Sharing System.” In *2012 15th International IEEE Conference on Intelligent Transportation Systems*. doi:10.1109/itsc.2012.6338703.
Osorio, Karel, Enrique Alba, and Gabriel Luque. 2013. “Using Theory to Self-Tune Migration Periods in Distributed Genetic Algorithms.” In *2013 IEEE Congress on Evolutionary Computation*, 2595–2601. IEEE.
Park, Taejin, Ri Choe, and Kwang Ryel Ryu. 2008. “Dual-Population Genetic Algorithm for Nonstationary Optimization.” In *Proceedings of the 10th Annual*

Conference on Genetic and Evolutionary Computation - GECCO '08, 1025. New York, New York, USA: ACM Press.

Park, Taejin, and Kwang Ryel Ryu. 2007. "A Dual Population Genetic Algorithm with Evolving Diversity." In *2007 IEEE Congress on Evolutionary Computation*, 3516–22. IEEE.

Popovici, Elena, and Kenneth De Jong. 2003. "Understanding EA Dynamics via Population Fitness Distributions." In *Genetic and Evolutionary Computation — GECCO 2003*, 1604–5. Springer, Berlin, Heidelberg.

Popper, K. R. 1999. *The Logic of Scientific Discovery*. NY: Routledge,.

Richter, Hendrik. 2009. "Detecting Change in Dynamic Fitness Landscapes." In *2009 IEEE Congress on Evolutionary Computation*. doi:10.1109/cec.2009.4983135.

Rohlfshagen, Philipp, Per Kristian Lehre, and Xin Yao. 2009. "Dynamic Evolutionary Optimisation." In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation - GECCO '09*, 1713–20. New York, New York, USA: ACM Press.

———. 2013. "Evolutionary Computation for Dynamic Optimization Problems." *Evolutionary Computation for DOPs* 490: 221–40.

Rohlfshagen, Philipp, and Xin Yao. 2008. "Attributes of Dynamic Combinatorial Optimisation." In *Simulated Evolution and Learning*, 442–51. Berlin, Heidelberg: Springer Berlin Heidelberg.

Shengxiang Yang. 2005. "Memory-Enhanced Univariate Marginal Distribution Algorithms for Dynamic Optimization Problems." In *2005 IEEE Congress on Evolutionary Computation*, 3:2560–67. IEEE.

Simões, Anabela, and E. Costa. 2007. "Improving Memory Usage in Evolutionary Algorithms for Changing Environments." In *2007 IEEE Congress on Evolutionary Computation*, 276–83. IEEE.

Simões, Anabela, and Ernesto Costa. 2011. "Memory-Based CHC Algorithms for the Dynamic Traveling Salesman Problem." In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation - GECCO '11*, 1037. New York, New York, USA: ACM Press.

Stanhope, Stephen A., and Jason M. Daida. 1998. "Optimal Mutation and Crossover Rates for a Genetic Algorithm Operating in a Dynamic Environment." In *Evolutionary Programming VII*, edited by V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, 693–702. Springer Berlin Heidelberg.

Talbi, El-Ghazali. 2009. *Metaheuristics: From Design to Implementation*. John Wiley & Sons.

Topcuoglu, Haluk Rahmi, Abdulvahid Ucar, and Lokman Altin. 2014. "A Hyper-Heuristic Based Framework for Dynamic Optimization Problems." *Applied Soft Computing* 19: 236–51.

Trojanowski, K., and Zbigniew Michalewicz. 1999. "Searching for Optima in Non-Stationary Environments." In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, 1843–50. IEEE.

Trojanowski, Krzysztof, and Zbigniew Michalewicz. 1999. "Evolutionary Approach to Non-Stationary Optimisation Tasks." In *Lecture Notes in Computer Science*, 538–46.

Ursem, Rasmus K. 2000. "Multinational GAs: Multimodal Optimization Techniques in Dynamic Environments." In *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation*, 19–26.

Wang, Hongfeng, Ilkyeong Moon, Shenxiang Yang, and Dingwei Wang. 2012. "A Memetic Particle Swarm Optimization Algorithm for Multimodal Optimization Problems." *Information Sciences* 197 (August): 38–52.
Weicker, Karsten. 2000. "An Analysis of Dynamic Severity and Population Size." In *Parallel Problem Solving from Nature PPSN VI*, 159–68. Springer Berlin Heidelberg.
Woldesenbet, Y. G., and G. G. Yen. 2009. "Dynamic Evolutionary Algorithm With Variable Relocation." *IEEE Transactions on Evolutionary Computation* 13 (3): 500–513.
Yang, Shengxiang. 2007. "Explicit Memory Schemes for Evolutionary Algorithms in Dynamic Environments." In *Computation in Dynamic and Uncertain Environments*, 28:3–28.
———. 2008. "Genetic Algorithms with Memory- and Elitism-Based Immigrants in Dynamic Environments." *Evolutionary Computation* 16 (3): 385–416.
Yang, Shengxiang, Yong Jiang, and Trung Thanh Nguyen. 2013. "Metaheuristics for Dynamic Combinatorial Optimization Problems." *IMA Journal of Management Mathematics* 24 (4): 451–80.
Yang, Shengxiang, and Changhe Li. 2010. "A Clustering Particle Swarm Optimizer for Locating and Tracking Multiple Optima in Dynamic Environments." *IEEE Transactions on Evolutionary Computation* 14 (6): 959–74.
Yaniasaki, K., K. Kitakaze, and M. Sekiguchi. 2002. "Dynamic Optimization by Evolutionary Algorithms Applied to Financial Time Series." In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, 2:2017–22. IEEE.
Yao, Xin, and Yong Xu. 2006. "Recent Advances in Evolutionary Computation." *J. Comput. Sci. & Tchnol.* 21: 1–18.
Yong, Cao, and Luo Wenjian. 2010. "A Novel Updating Strategy for Associative Memory Scheme in Cyclic Dynamic Environments." In *Third International Workshop on Advanced Computational Intelligence*, 32–39. IEEE.
Zheng, Bojin, Yuanxiang Li, and Ting Hu. 2006. "Vector Prediction Approach to Handle Dynamical Optimization Problems." *Proceedings of the 6th International Conference on Simulated Evolution And Learning*, 353–60.
Zhu, Tao, Wenjian Luo, and Zhifang Li. 2011. "An Adaptive Strategy for Updating the Memory in Evolutionary Algorithms for Dynamic Optimization." In *2011 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, 8–15. IEEE.
Zhu, Tao, Wenjian Luo, and Lihua Yue. 2014. "Combining Multipopulation Evolutionary Algorithms with Memory for Dynamic Optimization Problems." In *2014 IEEE Congress on Evolutionary Computation (CEC)*, 2047–54. IEEE.