

COMBINACIÓN MULTI-AGENTE DE ALGORITMOS EVOLUTIVOS Y MINERÍA DE DATOS PARA MEJORAR LA BÚSQUEDA EN PROBLEMAS DE OPTIMIZACIÓN DEL MUNDO REAL

Joaquín Izquierdo^{1*}, Enrique Campbell¹, Idel Montalvo² y Rafael Pérez-García¹

1: FluIng-IMM
Universitat Politècnica de València
Cno. de Vera s/n, 46022 Valencia, Spain
e-mail: {jizquier, encamgo1, rperez}@upv.es web: fluing.upv.es

2: Ingeniousware GmbH
Bahnhofstraße 4a, 76137 Karlsruhe, Germany
e-mail: imontalvo@ingeniousware.net web: ingeniousware.com

Palabras clave: Sistemas multi-agente, algoritmos evolutivos, minería de datos, sistemas de distribución de agua

Resumen *Los algoritmos evolutivos proporcionan una gran flexibilidad en la optimización ya que actualmente permiten el uso de prácticamente cualesquiera funciones objetivo y restricciones, aun cuando las evaluaciones requieran ejecutar complejas simulaciones matemáticas de los sistemas bajo análisis y, en particular, estén condicionados a prescindir del uso de derivadas. Sin embargo, seleccionar la heurística más apropiada para la resolución de un problema específico no es fácil ya que algunos algoritmos funcionan mejor en algunos problemas y peor o muy mal en otros. Las razones son múltiples. En primer lugar, algunos operadores se adaptan mejor a ciertos problemas que a otros. En segundo lugar, incluso cuando una población de soluciones evoluciona, la forma en la que lo hace no es lo suficientemente dinámica, dado el enorme tamaño del espacio a explorar en el caso de problemas del mundo real. En tercer lugar, tradicionalmente, los procesos de búsqueda de solución se han aplicado de manera indiscriminada a distintos tipos de problemas, ignorando su tamaño, complejidad y dominio. Esta contribución, propone un enfoque basado en la hibridación de varias metaheurísticas que actúan sinérgicamente, en el uso de parámetros auto-adaptativos, y en la introducción de reglas obtenidas tanto del conocimiento del problema como derivadas del descubrimiento de conocimiento (técnicas de minería de datos) a partir de bases de datos de soluciones exploradas en las generaciones anteriores. El entorno general para estos ingredientes es el llamado paradigma multi-agente. Con la combinación de estos elementos, se pueden desarrollar aplicaciones de apoyo a la toma de decisiones. Este trabajo busca ayudar en la formulación del problema y aumentar la eficiencia computacional para poder tratar problemas del mundo real. En concreto, aquí abordamos el problema del diseño óptimo de una red de distribución de agua.*

1. INTRODUCCIÓN

Las razones por las que muchos problemas de optimización en la ingeniería trascienden actualmente la categoría de lo que podría llamarse optimización estándar [1-2] son múltiples. Por nombrar sólo algunas de tales razones, los problemas presentan no linealidades, carecen de buenas condiciones de diferenciabilidad, son multimodales, y están condicionados por diversos factores difíciles de manejar. Para estos problemas, la optimización global es, a menudo, no factible. Además, el ruido y la incertidumbre inherente a los problemas del mundo real afectan a los métodos numéricos utilizados para evaluar los objetivos de la optimización y las restricciones.

No hay duda de que para permitir el mayor número de criterios de optimización y de restricciones, y manejar apropiadamente las singularidades (no linealidad, coexistencia de variables mixtas, etc.) asociadas se necesitan nuevos métodos. En general, los problemas del mundo real tienen que ser abordados mediante técnicas de optimización adecuadas en combinación con técnicas de simulación y análisis de datos.

Los algoritmos evolutivos (AEs), que son técnicas de optimización estocásticas que evitan diversas complicaciones matemáticas [3], manejan poblaciones de soluciones, tratando de identificar el o los mejores individuos que representan la o las mejores soluciones para el problema. La literatura es muy extensa en ejemplos en diversos campos de la ingeniería, y más concretamente en el sector del agua [4] y, en particular, en hidráulica urbana [5] (campo de especialización de los autores, con respecto al diseño, la calibración, el ahorro de energía, etc.). Véanse, entre otros muchos [6-15]. La flexibilidad introducida por los algoritmos evolutivos permite la utilización de prácticamente cualquier función objetivo para evaluar las soluciones, incluso cuando estas evaluaciones requieran ejecutar complejas simulaciones matemáticas y / o procedurales de los sistemas bajo análisis.

Sin embargo, cada algoritmo se adapta mejor a unos problemas que a otros, puesto que la heurística detrás de un determinado algoritmo evolutivo dota a sus elementos de capacidades específicas para resolver eficientemente algunos tipos de problemas, mientras que se muestra ineficiente con problemas de distinta naturaleza [16].

Se impone realizar mejoras claras en los AEs para ser eficientes en la solución de problemas reales.

Como objetivo general, tratamos de innovar en calidad en la búsqueda de soluciones en problemas de optimización complejos utilizando una metodología (no estándar) de optimización que se basa en el paradigma de los sistemas multi-agente (MASs, por *multi-agent systems*), e integra diversas heurísticas, parámetros auto-adaptativos, y elementos portadores de información basados en el conocimiento específico de los problemas y en la explotación de las soluciones analizadas. Esta metodología también deberá permitir el control y la interacción de uno o más usuarios como agentes de alto nivel en el sistema, de modo que el proceso de optimización sea interactivo, y pueda, por lo tanto, ser conducido de manera más eficiente.

Una plataforma computacional ya construida por los autores, que utiliza algunas de estas ideas, se ha modificado y ampliado de manera adecuada con la incorporación de los conocimientos obtenidos durante el proceso de optimización, para ofrecer la posibilidad de

explorar el comportamiento emergente de alto nivel en los procesos de optimización basados en conocimiento obtenido de la interacción de bajo nivel.

Este documento se estructura como sigue. En la siguiente sección se presenta el enfoque multiagente, que envuelve todos los ingredientes de nuestra propuesta, algunos de los cuales se describen sucintamente, a saber, la hibridación, la auto-adaptabilidad de los parámetros, y el uso de reglas dependientes del problema. En la Sección 3 se razona que otros tipos de conocimiento extraídos de la historia de la evolución también pueden ser útiles, y nos centramos en un método concreto de minería de datos (DM), a saber, los mapas auto-organizados de Kohonen (SOMs, del inglés *self-organizing maps*). La sección 4 presenta un caso de estudio. Finalmente, la sección 5 presenta las conclusiones.

2. LA ESTRUCTURA MULTI-AGENTE

La estructura que envuelve nuestra propuesta se basa en el uso de agentes [17-18]. Los MASs han proporcionado una base teórica y computacional adecuada para la solución de diversos problemas en diferentes campos.

Además de ASO [15], desarrollado por los autores, varias publicaciones han considerado la optimización para ciertos problemas complejos basada en agentes. Los ejemplos incluyen [19], que desarrolla un sistema de optimización multi-agente para problemas de programación; [20], que utiliza una combinación de MASs con técnicas de optimización en un problema de distribución dinámica de asignación de recursos; y [21], que utilizan MASs en optimización combinatoria, entre otros.

En un MAS, cada agente tiene una capacidad limitada y / o información incompleta para resolver un problema y, por lo tanto, tiene una visión limitada de la solución. No existe un control general del sistema, los valores están descentralizados y el cálculo es asíncrono [17]. Cada agente, que actúa por sí solo, no puede resolver el problema en su totalidad. Sin embargo, un grupo de agentes, aprovechando la coexistencia de diferentes puntos de vista, es más capaz de encontrar una solución como resultado de la interacción entre los agentes. Esta idea puede ser claramente extrapolable al caso de la optimización, ya que el resultado de las múltiples interacciones que ocurren dentro de un MAS mejora el rendimiento. Asociaciones de agentes interactivos dan lugar a estructuras colectivas, llamadas *swarms* en [15], que representan un comportamiento colectivo. Estas estructuras también pueden ser considerados como agentes a un nivel de abstracción superior, y tienen su propio comportamiento. A su vez, pueden interactuar con otros *swarms*. El comportamiento general del sistema surge, por lo tanto, a partir del comportamiento local de los agentes.

Los MASs favorecen el diseño e implementación de sistemas distribuidos de manera que se acelera la búsqueda de nuevas soluciones. Los beneficios asociados al uso de un enfoque basado en agentes, en comparación con los enfoques tradicionales de computación, son importantes en especial, en los casos en que la solución del problema exige la interacción entre los distintos participantes en el proceso, incluidos los participantes humanos. Según Wooldridge [18], "la interacción es probablemente la característica más importante del software complejo". El diseño de la conducta autónoma y emergente y que, por lo tanto puede ser naturalmente formalizado y, en consecuencia, automatizado es un ingrediente básico en

los MASs. Como resultado, los agentes adaptan su comportamiento de una manera orientada a eventos, que es básica para la solución de problemas de optimización complejos. En el siguiente apartado se describe el resultado de una integración de la optimización en un MAS, sobre la que vamos a construir el enfoque presentado en esta contribución. Los diferentes conceptos que se presentan en este documento se describen de forma concisa en los apartados siguientes.

2.1. La plataforma ASO

ASO [15] significa optimización basada en *swarms* de agentes (por *agent swarm optimization*, en inglés). La idea inspiradora de ASO fue un entorno basado en PSO (*particle swarm optimization*) desarrollado por los autores, que tuvo como objetivo imitar el juicio del ingeniero [10]. Fue construido utilizando varias características previas y ciertas mejoras sobre la inteligencia de los *swarms*. Los sistemas multi-agente, y la necesaria adaptación a problemas multi-objetivo, incluida la interacción humana, también están integrados en ASO.

ASO se implementa en un paquete de software llamado WaterIng [16], que fue desarrollado para el diseño de sistemas de distribución de agua, aunque su arquitectura permite la solución de problemas generales de optimización.

La aplicación de ASO a los problemas de benchmarking más populares de la literatura sobre diseño de sistemas de distribución de agua (SDAs) ha obtenido las mejores soluciones conocidas para estos problemas [16]. Los autores también han utilizado este paquete para abordar otros problemas complejos del mundo real, tales como el diseño de sistemas de aguas residuales [22]; la calibración de SDAs [8]; el diseño óptimo de una cadena de suministro de biomasa a nivel regional [23]; el clustering de una base de datos de una compañía de agua para clasificar las tuberías teniendo como objetivo la rehabilitación [24]; y la sectorización de SDAs [25].

2.2. Hibridación de AEs

ASO integra diversos algoritmos en tiempo de ejecución en una sola plataforma. La mezcla de diferentes algoritmos y la incorporación de nuevos agentes en tiempo de ejecución dentro de ASO son posibles porque ASO hace uso de computación paralela y distribuida para permitir la incorporación de nuevos agentes, así como el comportamiento asíncrono de los agentes. En la versión actual de ASO, los algoritmos se añaden manualmente por el usuario.

Sin embargo, al ampliar esta perspectiva con el uso de una estrategia multi-agente, la decisión acerca de la metaheurística más apropiada y / o la combinación híbrida de metaheurísticas quedaría en manos de un agente no humano especializado (un director) que podría tomar el papel del experto(s) con respecto a esta decisión específica. Este director se especializaría en el lanzamiento del AE o AEs más adecuados para el problema. Se puede considerar como un agente experto del sistema guiado por un sistema de apoyo a la decisión sobre el problema o campo específico, y alimentado por una base de datos o un repositorio de conocimiento de los problemas y casos particulares de problemas, adecuadamente mantenida por los expertos humanos. El director estaría a

cargo de la implementación de los AEs más adecuados para el tipo de problema a resolver, y del lanzamiento de nuevos *swarms* para cubrir o explotar áreas específicas del espacio de decisión. De esta manera, el proceso de hibridación quedaría libre de la subjetividad derivada de la experiencia de usuario, y podría ser sistematizado. Así, los agentes se adaptan a su capacidad para la exploración y explotación de una manera orientada a eventos que permite una aproximación mejor al campo específico de la problemática considerada y / o al caso del problema actual a resolver. Este enfoque permite la selección del algoritmo (o combinación de algoritmos) más adecuado para encontrar la mejor solución para el problema específico, y se utiliza como un concepto de alto nivel para decidir cómo se hibridan diversas estrategias de optimización de forma flexible y consistente, esperando que puedan encontrarse mejores soluciones mediante la cooperación.

La combinación de varios *swarms* dentro del mismo algoritmo es eficiente porque lleva a cabo una búsqueda local en la que cada uno de los *swarms* se especializa, y se utilizan los buenos elementos de mejora en términos de óptimo de Pareto para producir una nueva solución. La práctica de la incorporación de diferentes mecanismos de búsqueda también reduce la probabilidad de que la búsqueda quede atrapada en óptimos locales.

La implementación es factible utilizando las capacidades de Microsoft.Net Framework 4.0 para ejecutar diferentes instancias de *swarms* en paralelo y sincronizando su trabajo. La computación distribuida se basa en las capacidades de Windows Communication Foundation (incluido en Microsoft.Net Framework) para comunicar y sincronizar instancias de *swarms* que se pueden ejecutar en diferentes procesos / máquinas.

2.3. Parámetros auto-adaptativos

En la mayoría de los casos, los parámetros sobre los que se basan los AEs se definen y se fijan con antelación. El proceso de puesta a punto de un conjunto de parámetros de éxito es generalmente tedioso y lento. Sin embargo, existen métodos que utilizan parámetros adaptativos y auto-adaptativos; por ejemplo, los algoritmos ASO [15] y TRIBES [26] se basan en versiones sin parámetros de PSO; en [27] una máquina de vectores soporte fue entrenada para generar parámetros de PSO mientras se exploraba el espacio de soluciones de un problema. Otros algoritmos auto-adaptativos incluyen [28-31]. A pesar de estos intentos, muchos métodos de optimización recientes (incluyendo sus variantes) siguen utilizando parámetros que se ajustan *a priori* y, con frecuencia, tras procesos muy costosos.

En el caso de los parámetros auto-adaptativos, tales parámetros se incorporan en la representación de la solución. Mediante la introducción de los parámetros en los mecanismos internos de un EA, los individuos serán dotados, además, con la posibilidad de ajustar sus parámetros apuntando tanto a los parámetros que tenían cuando llegaron a su mejor estado previo, como a los parámetros de los mejores individuos que los llevaron a su posición privilegiada. Como consecuencia, los individuos no sólo utilizan la cooperación social y la cognición individual para mejorar su satisfacción del objetivo, sino también para mejorar la forma en la que lo hacen, al acomodarse a las mejores condiciones conocidas: a saber, las condiciones que, hasta el momento, han permitido a

los mejores individuos [32] su grado de adecuación al objetivo. Como valor añadido, el esfuerzo manual que se hace normalmente para inicializar nuevas ejecuciones queda estratégicamente en manos de los agentes. Como resultado, el número de variables se incrementa en el número de parámetros auto-adaptativos. Esto representa un inconveniente mínimo en comparación con la reducción en el espacio de búsqueda se deriva del uso de reglas, como se propone en este documento.

2.4. Reglas dependientes del problema

Sin embargo, el paso importante no vendrá del resultado de la combinación de varias heurísticas, o del mejor ajuste de parámetros, sino de influir más directamente la forma en que se realiza la búsqueda. Notamos primero que recientemente se han realizado una serie de sugerencias que abordan la necesidad de construir métodos de optimización que ofrezcan simultáneamente propiedades de distribución, cooperación y adaptación (por ejemplo, [33-37,15]). La idea es lograr un enfoque integral para la optimización basada en la simulación, que también incluya técnicas de Inteligencia Artificial (AI, por sus siglas en inglés), tales como los MASs, y sistemas expertos basados en el conocimiento en la búsqueda. El objetivo es crear, junto con técnicas computacionales apropiadas y la simulación, un entorno más favorable para la optimización.

En efecto, la metodología MAS ofrece el concepto general de la forma de organizar y diseñar un entorno de este tipo - con las interacciones adecuadas. El componente basado en el conocimiento, a su vez, puede ser usado para definir el comportamiento de ciertos agentes estratégicos. Estos dos conceptos se complementan entre sí y forman la base para el diseño de una propuesta definida por el enfoque basado en el agente. Además, el paradigma MAS modela el entorno mediante coordinación descentralizada. La integración de conocimiento en el entorno se distribuye a través de los agentes, lo que reduce la complejidad de modelar el proceso de búsqueda de solución.

Nuestro objetivo es desarrollar algoritmos que adapten su comportamiento a los problemas que se resuelven, para que tengan más posibilidades de éxito. Una forma de lograrlo es mediante la combinación de la optimización evolutiva con la introducción de, entre otros portadores de conocimiento, reglas e información de probabilidad de búsqueda basadas en el dominio del problema a resolver.

Para cumplir con el tercer ingrediente principal de nuestro enfoque, este trabajo sugiere la aplicación de técnicas de DM al conjunto de soluciones evaluadas después de varias generaciones de una ejecución evolutiva, con el fin de obtener conocimiento para ser utilizado por las generaciones futuras de manera que se mejore la búsqueda.

Dentro de la metodología multi-agente, la acción de los expertos (humanos) en los procesos de optimización también puede ser explotada - aunque sólo parcialmente - mediante agentes (artificiales) que actúen como sustitutos de los humanos utilizando conocimiento sintético que puede ser definido mediante información derivada específicamente de un buen conocimiento del problema. Esta información trata de imitar el juicio de un experto humano al considerar la solución a un problema. Los algoritmos evolutivos en general no se han aprovechado de esta posibilidad y, así, han tenido que analizar espacios de búsqueda más grandes de lo necesario.

Incluir reglas puede reducir el espacio de búsqueda en varios órdenes de magnitud [16]. Esta medida también puede verse como una forma de mejorar la definición del problema mediante la inclusión de información que, o bien no se puede expresar con facilidad, o depende de la solución actual.

Como consecuencia del uso de esta funcionalidad, las soluciones son, a la vez, eficientes y más cercanas a la realidad. La eficiencia deriva del hecho de que, simplemente seleccionando algunas reglas simples, por lo general, se evitan muchos cálculos o simulaciones caros (simulaciones hidráulicas en el estudio de caso que presentamos en este trabajo). Y el hecho de que tal información tiene un fuerte significado dependiente del problema, definitivamente acerca la solución a la realidad. La especificación de la información depende fuertemente del problema en consideración.

Utilizamos aquí el problema de diseño de SDAs para dar un ejemplo. En [15-16] se utilizaron agentes basados en reglas para la solución de problemas relacionados con el diseño de los sistemas de distribución de agua. Para el dimensionamiento de las tuberías en un SDA, una regla básica es reducir o mantener el diámetro de los tubos conforme el sistema progresa de aguas arriba a aguas abajo. La implementación de esta regla en los agentes permitió diseños con fiabilidad mejorada. La no inclusión de esta regla hace que algunos algoritmos evolutivos obtengan soluciones no elegantes desde el punto de vista de la ingeniería, por ejemplo, para redes de distribución de tamaño grande.

La idea es que, durante la búsqueda, los rangos de ciertas variables de decisión sean modificados dinámicamente (reducidos, en general) dependiendo de los valores obtenidos por otras variables de decisión.

La información considerada hasta aquí se deriva exclusivamente del conocimiento experto del problema considerado. En el párrafo siguiente explotamos esta idea para el caso de conocimiento sintético.

3. DESCUBRIMIENTO DE CONOCIMIENTO PARA MEJORAR LA BÚSQUEDA

Aunque las ideas anteriores funcionan bastante bien, tienen un par de inconvenientes. En primer lugar, la información deben ser "codificada": su implementación supone cambiar el código fuente existente o agregar nuevo código. En segundo lugar, puede ser difícil, en algunos campos y/o problemas, descubrir nueva información para ayudar a mejorar el proceso de búsqueda. El desarrollo de agentes basados en reglas dentro de los AEs para mejorar su desempeño requiere la participación activa de especialistas del dominio del problema. Es difícil desarrollar buenas reglas sin una buena comprensión de los problemas en el contexto de su dominio. Pero incluso para las personas con un profundo conocimiento del dominio del problema es difícil definir reglas que puedan generalizarse y aplicarse para trabajar en combinación con técnicas evolutivas. Pensamos que es mucho más fácil analizar la forma de mejorar la búsqueda en el caso de un problema específico que definir una manera general de hacerlo. Incluso si se encuentra una forma general, será necesario ajustarla para que pueda ser expresada en un lenguaje de programación de modo que pueda ser utilizada eficientemente.

En esta sección argumentamos que, además de las características anteriores, el proceso de

búsqueda debe estar lo más cerca posible al problema específico que se está solucionado, ya que es razonable pensar que los algoritmos que sean capaces de adaptar su comportamiento al problema considerado tendrán más posibilidades de éxito. Esta idea podría ser abordada mediante la combinación de la forma de trabajo propia de los algoritmos evolutivos con la introducción de conocimiento extraído de una base de datos adecuada de soluciones visitadas durante los pasos anteriores del proceso de optimización. Desarrollamos esta idea en esta sección.

Durante la ejecución de los AEs, por lo general, el número de soluciones evaluadas representa un pequeño porcentaje del espacio total de la solución que corresponde al problema a resolver. Sin embargo, el número de soluciones evaluadas suele ser considerable, y la mayoría de técnicas evolutivas utilizan sólo una pequeña porción a la vez. Muchas de las soluciones evaluadas durante el proceso de búsqueda son "olvidadas" tras una o varias generaciones, y la experiencia combinada de las generaciones anteriores no se explota.

Las técnicas de DM pueden permitir una visión más detallada de las muchas soluciones "buenas" que se han considerado y que han sido rápidamente descartadas porque estaban dominadas por mejores soluciones en un momento efímero en el proceso de la evolución. Utilizando bases de datos obtenidas mediante el registro adecuado de esas soluciones ignoradas, las técnicas de DM pueden ayudar a entender mejor y describir cómo un sistema puede reaccionar o comportarse tras la introducción de cambios.

Como se ha dicho más arriba, la propuesta principal de este trabajo es la integración de técnicas de DM en el trabajo evolutivo como un paso para la generación dinámica de conocimiento que puede ser utilizado para mejorar la eficiencia de los procesos de búsqueda de solución.

Describimos ahora una tal integración. Al inicializar un proceso evolutivo sólo hay, generalmente, disponibles soluciones aleatorias. Por lo tanto, al principio, hasta cierto punto en la iteración, el algoritmo utiliza sus propios mecanismos de búsqueda, tal vez con la ayuda de algunas reglas específicas claras del problema. Mientras tanto, (algunas de) las soluciones analizadas se almacenan en una base de datos adecuada. En un cierto momento de la iteración, se pone en marcha el algoritmo de extracción de conocimiento, generando una serie de pautas o reglas. Este nuevo conocimiento se codifica automáticamente. Durante un nuevo lote de iteraciones se aplican estas reglas, mientras se genera una nueva base de datos. Después de esta iteración, se pone en marcha un nuevo proceso de extracción de conocimiento, que tal vez puede tomar ventaja de las viejas reglas. Entonces se aplica el nuevo conocimiento y se repite el proceso hasta la convergencia. Esto permite acelerar la convergencia.

Antes de descender a especificaciones concretas, vamos a señalar una serie de aspectos que influyen en este proceso. Daremos ejemplos específicos al presentar el caso de estudio en la Sección 4.

En primer lugar, si existen reglas claras bien definidas dependientes del problema deben ser codificadas, para ser aplicadas desde el primer momento, forzando así al proceso a producir soluciones más próximas a la realidad.

En segundo lugar, los puntos en la iteración en los que el proceso evolutivo puede ser detenido para proceder a la extracción de conocimiento se pueden decidir de varias maneras; por ejemplo, después de un número fijo de iteraciones, o cuando se haya conseguido una

mejora significativa que se estabiliza.

En tercer lugar, para una base de datos típica, puede ser que algunas de las variables sean irrelevantes para el descubrimiento de nuevas reglas ya que presentan un valor casi constante en un porcentaje muy alto de los registros. En una determinada etapa de la evolución, estos valores "constantes" pueden corresponder, bien a los valores óptimos (objetivo) o a variables que no han sido completamente exploradas. La decisión debe tomarse de acuerdo a la etapa de la evolución. En etapas tempranas de la evolución esos valores serán simplemente ignorados, ya que muy probablemente corresponden a soluciones poco exploradas, correspondiendo así a mínimos locales desde donde un EA debería utilizar sus habilidades estocásticas para escapar. En cambio, en etapas de evolución avanzada, estos valores deben ser transformados directamente en reglas duras para esas variables, si las soluciones corresponden a buenas soluciones actuales. En cualquier caso, estas variables pueden ser (temporalmente) eliminadas de la base de datos para la situación actual.

Diversas técnicas de minería de datos pueden ser utilizadas para extraer conocimiento de las bases de datos de soluciones analizadas previamente. En este artículo se analizan los SOMs, mapas autoorganizados de Kohonen [38].

3.1. Zonas de búsqueda más probable: mapas de Kohonen

Los SOMs de Kohonen son conocidos como un paradigma importante de red neuronal no supervisada de análisis de datos [38]. El algoritmo de aprendizaje sigue el patrón de los modelos competitivos, pero la regla de actualización produce una capa de salida en la que se conserva la topología de los patrones de entrada. Esto significa que si dos patrones están cerca en el espacio de entrada (en el sentido de alguna medida de similitud, como las medidas utilizadas en las estrategias denominadas *the winner takes all*) también sus correspondientes neuronas activas están topológicamente cerca en la capa de salida. Una red que realiza esta función se llama un mapa de características. Estos mapas no sólo agrupan patrones de entrada en clústeres, sino que también describen visualmente la relación entre los clústeres del espacio de entrada.

Un mapa de Kohonen es una matriz bidimensional de neuronas que están totalmente conectadas con el vector de entrada y organizadas en un cuadrado o un hexágono. Una disposición hexagonal es más aconsejable, porque al final del proceso de aprendizaje se tiene una mejor visualización de la estructura del espacio de entrada.

Los elementos del input del mapa constituyen una base de datos de n -dimensional de vectores X_1, X_2, \dots, X_m . Las componentes de estos vectores son el input para la red. En el entrenamiento secuencial, en cada paso de aprendizaje, un vector de entrada $X_p \in \{X_1, X_2, \dots, X_m\}$ es presentado a la red neural. Vectores reales n -dimensionales de otro conjunto, $\{M_i\}$, representan las aproximaciones calculadas sucesivamente de los modelos (neuronas) M_i . Aquí i es el índice espacial del nodo de la matriz con el que se asocia el modelo M_i . El vector X_p se compara con todos los modelos M_i . Por lo general, se utiliza la distancia euclídea $\|X_p - M_i\|_2$ entre el vector de entrada X_p y cada modelo M_i para esta comparación. El modelo M_c con la mínima distancia euclídea al X_p es designado como ganador. Las coordenadas de los modelos se adaptan de acuerdo a la regla de aprendizaje:

$$\Delta M_i(t) = \eta(t)H(c,i,t)(X_p - M_i), \quad (1)$$

donde $\eta(t)$ es una función escalar de t monótonamente decreciente, t es el paso de la iteración, y $H(c,i,t)$ se llama la función de vecindad. Esta función se asemeja a la del núcleo que se aplica en las funciones de base radial [39] y es la encargada de controlar el radio alrededor del modelo ganador, que permite producir una actualización gradual. El valor de c es el índice de un modelo concreto, el ganador, $M_c(t)$ en la matriz, a saber, el que está más cerca de $X(t)$ (el X_p actual):

$$c = \arg \min_i \|X(t) - M_i(t)\|_2.$$

El algoritmo SOM asume que este proceso converge y produce valores ordenados para los modelos deseados. Hay varias opciones posibles para la forma matemática de $H(c,i,t)$. Una de estas opciones es muy simple: $H(c,i,t) = 1$ para cierto radio alrededor de la ganadora, y cero en caso contrario. Una opción mucho más utilizada para la función de vecindad $H(c,i,t)$ es la de gaussiana

$$H(c,i,t) = e^{-\frac{\|M_i - M_c\|_2^2}{2\sigma^2(t)}},$$

donde $\sigma(t)$ es otra función monótonamente decreciente de t (una varianza que controla la anchura de la gaussiana y, por tanto, el radio alrededor del modelo ganador), que es bastante grande en el comienzo del proceso (de alrededor de la mitad a dos tercios del diámetro de la matriz), y se reduce gradualmente a una fracción de este tamaño durante la primera parte del proceso de iteración. Según Kohonen [39], el orden topológico se desarrolla durante este período. Después de esta fase inicial de ordenación en bruto, se produce la convergencia final. Cuando se utiliza el entrenamiento por lotes (*batch*), todo el conjunto de muestras se presenta a la red y se obtienen modelos ganadores; después de esto, los modelos se actualizan con el efecto de todas las muestras:

$$M_i = \frac{\sum_j n_j \eta H(j,i) X_{m,j}}{\sum_j n_j \eta H(j,i)}.$$

Aquí $X_{m,j}$ representa la media de las entradas que están más cerca del modelo M_j , y n_j es el número de tales entradas.

Tras el entrenamiento de la red SOM, se debe evaluar su calidad. El error denominado de *quantization* muestra lo bien que las neuronas de la red entrenada se adaptan a los vectores de entrada. Este error es la distancia media entre los vectores de datos X_p a sus neuronas ganadoras $M_c(p)$:

$$Q = \frac{1}{m} \sum_{p=1}^m \|X_p - M_{c(p)}\|_2^2. \quad (2)$$

La propiedad de preservar la topología se obtiene mediante la regla de aprendizaje (1) que

involucra a la neurona ganadora y a sus vecinas en el proceso de actualización. Las neuronas cercanas aprenden a activarse cuando se les muestran patrones similares. Durante el entrenamiento, la red asigna a las neuronas una posición en el mapa basada en la característica dominante del patrón de entrada. Por esta razón, los mapas de Kohonen son llamados mapas auto-organizados.

Si el espacio de entrada es altamente dimensional, los mapas de Kohonen se pueden interpretar como proyectores en una matriz bidimensional de neuronas que tiene en cuenta la densidad de probabilidad de los datos y preserva la topología del patrón de entrada original. La preservación de la topología del patrón de entrada original es una gran ventaja para la visualización de resultados [40-41].

Nuestra idea es utilizar esas proyecciones para obtener perspectivas bidimensionales del paisaje de los objetivos estudiados, de modo que puedan identificarse áreas probables de buenas soluciones que pueden proporcionar al proceso evolutivo pistas interesantes acerca de dónde se encuentran. Sin embargo, en los problemas de optimización, las bases de datos obtenidas durante la evolución, contienen dos tipos de variables, variables independientes (decisión) y variables dependientes (objetivos). De alguna manera, el estudio de estos datos se parece más a un proceso supervisado que a uno no supervisado.

La descripción anterior se refiere a la idea básica de SOM (sin supervisión). Sin embargo, según lo sugerido por Kohonen [38], las redes supervisadas de Kohonen (SKN, por *supervised Kohonen networks*), basadas en el caso sin supervisión, son una alternativa más potente de modelado. En las SKNs, hay dos tipos de objetos: una variable independiente - objeto X - (vector o diámetros de las tuberías en nuestro caso de estudio) y una variable dependiente - objeto Y - (coste de la solución). En las SKNs, durante el proceso de actualización ambas variables se acoplan, generando un mapa concatenado, y, después de este proceso de formación, se desacoplan.

Hay dos problemas importantes asociados con las SKNs. En primer lugar, las variables de los objetos X e Y en el conjunto de entrenamiento deben ser escaladas adecuadamente a fin de lograr una inmersión óptima de la topología del espacio de entrada y de salida en el mapa concatenado de la red SKN. En segundo lugar, considerar el peso relativo del número de variables en X y del número de variables en Y durante la formación de una SKN no es trivial. El algoritmo *XY fused network* (XYF) [42] permite hacer frente a los problemas mencionados anteriormente. Este algoritmo encuentra un modelo ganador común que está determinado por la localización de un valor mínimo denominado *Fused Similarity Measure*, $S_{\text{Fused}}(i,k)$ (véase la ecuación (3) a continuación). Esta medida se calcula mediante la combinación ponderada de las similitudes entre un objeto X_i y todos los modelos, MX , en el X_{map} , $S(X_i, MX)$, y las similitudes entre el correspondiente objeto de salida Y_i y los modelos, MY , en el Y_{map} , $S(Y_i, MY)$. Para tratar las posibles diferencias en la magnitud de estas medidas de similitud $S(X_i, MX)$ y $S(Y_i, MY)$, ambos conjuntos de similitudes son re-escalados utilizando su valor máximo, de modo que las distancias máximas tanto en X como en Y sean igual a 1:

$$S_{\text{Fused}}(i,k) = \alpha(t)S(X_i, MX_k) + (1-\alpha(t))S(Y_i, MY_k). \quad (3)$$

Aquí $\alpha(t)$ es un parámetro que regula el peso relativo entre las similitudes $S(X_i, MX_k)$ y $S(Y_i, MY_k)$, y t en $\alpha(t)$ de nuevo se refiere al número de iteración (expresado en *epochs*) durante

el entrenamiento. En la ecuación (3), un *epoch* equivale a la presentación a la red XYF de todos los objetos X (y por ende Y) contenidos en el conjunto de entrenamiento. Para el entrenamiento XYF, $\alpha(t)$ disminuye linealmente en el tiempo, lo que implica que en la etapa inicial del entrenamiento la similitud entre los objetos X y los modelos de la X_{map} dominará la determinación del modelo ganador común. Al final del entrenamiento, ambas similitudes $S(X, X_{\text{map}})$ y $S(Y, Y_{\text{map}})$ contribuyen igualmente a la determinación del modelo ganador compartido.

Para mostrar la idoneidad de los SOMs para detectar áreas de mejor probabilidad de búsqueda en el proceso de optimización considerado en el caso de estudio de la Sección 4, se utiliza la aplicación en R de la función XYF por Wehrens y Buydens [43]. Hacemos notar aquí que la función *supersom*, implementada en el mismo paquete, permite la consideración de diversas variables dependientes y es adecuada para problemas de optimización multi-objetivo.

4. CASO ESTUDIO

En este párrafo se aplican las ideas anteriores al diseño de una RDA. Estudiamos la red Hanoi para obtener áreas de búsqueda con mejor probabilidad a través de los SOMs. También se discuten los resultados obtenidos.

4.1. Red de Hanoi

El problema de distribución de agua de Hanoi se ha discutido ampliamente en la literatura [44-49], entre muchos otros. Para medir la efectividad de nuestro algoritmo propuesto, vamos a aplicarlo a este mismo problema. La red consta de una fuente de altura piezométrica fija, 34 tuberías y 31 nodos de demanda sujetos a una condición de carga. La red tiene tres mallas y dos ramificaciones. La figura 1 contiene una representación de la red.

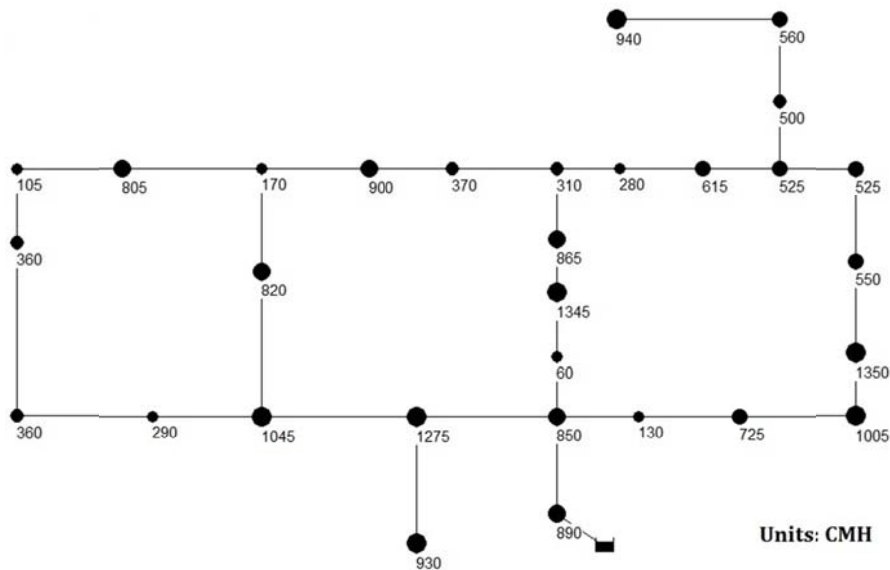


Figura 1. La red de Hanoi

Los números próximos a los nodos representan el consumo asociado a los nodos en m^3/h (también proporcionales al tamaño de los nodos). Se trata de encontrar los diámetros de los 34 tubos de tal manera que el coste total de esta red sea mínimo y la presión en cada nodo de consumo sea de al menos 30 m. La configuración completa se puede encontrar, entre otros muchos lugares, en [50].

Para comprobar el proceso descrito más arriba, se puso en marcha el siguiente experimento varias veces, a partir de diferentes inicializaciones, con resultados muy similares. Un *swarm* de ASO con una población de 100 individuos inició el proceso. Durante el primer lote de iteraciones, se generó una base de datos de soluciones. Después de este lote, una red con una topología hexagonal de 5×8 neuronas fue entrenada - utilizando la base de datos compilada usando la función XYF con los diámetros de tubería, que son las variables independientes, y el coste, que es la variable dependiente. En este punto, el *swarm* se duplicó y los dos *swarms* idénticos se pusieron a competir en paralelo para el problema de Hanoi. Uno de los *swarms* utilizó PSO auto-adaptativo más la regla *a priori* que impone diámetros más pequeños aguas abajo. El otro *swarm* siguió el proceso descrito en el documento, por lo que utilizó, además, el conocimiento descubierto. Para este último *swarm*, la iteración continuó y se obtuvo pronto una importante mejora que fue consolidada. La condición de terminación detuvo el proceso para este *swarm*. El otro *swarm* continuó iterando hasta alcanzar la convergencia más tarde que el primero. Proporcionamos los detalles a continuación.

En este problema, treinta y cinco columnas constituyen los campos de la base de datos, que corresponden a los valores del diámetro de cada una de las 34 tuberías de la red, más el valor objetivo, correspondiente al coste de la red incluida la penalización incurrida por no satisfacer el valor de presión mínima de 30 m. Los diámetros candidatos fueron codificados usando números de 0 a 5, y el objetivo fue discretizado en cuatro categorías para obtener un atributo de coste cualitativo. La primera categoría incluye las soluciones excelentes, lo que corresponde a los registros con objetivos entre 0 y 3% sobre la solución más barata en la base de datos; las soluciones buenas, incluyen los registros con objetivo entre 3% y 5% sobre la solución más barata; las soluciones pobres son las que tienen un costo de entre 5% y 15% sobre la solución más barata; y, finalmente, el resto de las soluciones constituye la clase de las soluciones malas.

La Figura 2 muestra el SOM obtenido después de la primera tanda de iteración para una de las ejecuciones. En este SOM, las soluciones económicas actuales se encuentran en las neuronas en la parte inferior izquierda del SOM, como se muestra en la primera capa (que reúne a casi toda las soluciones excelentes y buenas en la base de datos actual). La neurona que reúne la mayoría de las soluciones excelentes actuales corresponde a la neurona más abajo y a la izquierda: obsérvese, mirando la segunda capa, el gran número de excelentes soluciones concentradas en esta neurona. Hay que tener también en cuenta la calidad de este clúster. En efecto, la tercera capa representa, para cada neurona, la distancia media, véase la ecuación (3), entre cada muestra asignada a la neurona y el *codebook* de la neurona: cuanto menor es el valor asociado, mejor es la identificación de las muestras asignadas con el *codebook*. Podemos observar cómo la neurona inferior izquierda, que contiene a la mayor parte de las soluciones excelentes, exhibe un valor bajo para la distancia media, lo que revela que el *codebook* correspondiente representa adecuadamente esas muestras. El *codebook* para esta

neurona está dado por los valores de los cuadrados a la izquierda de cada par en la Figura 3.

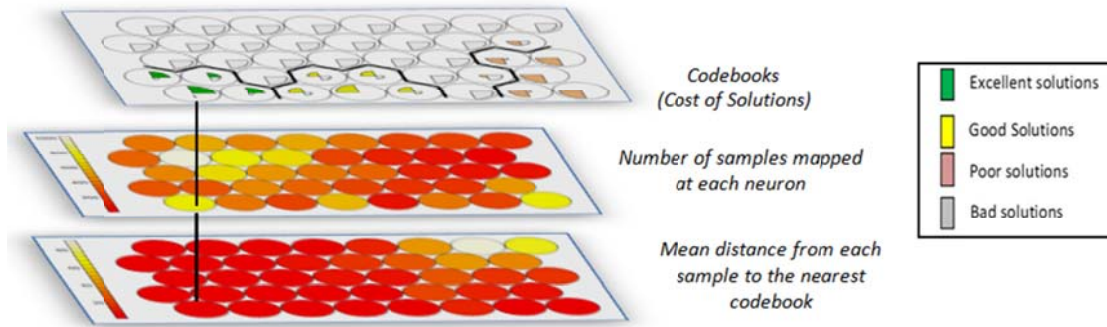


Figura 2. SOM tras la primera parte de la iteración

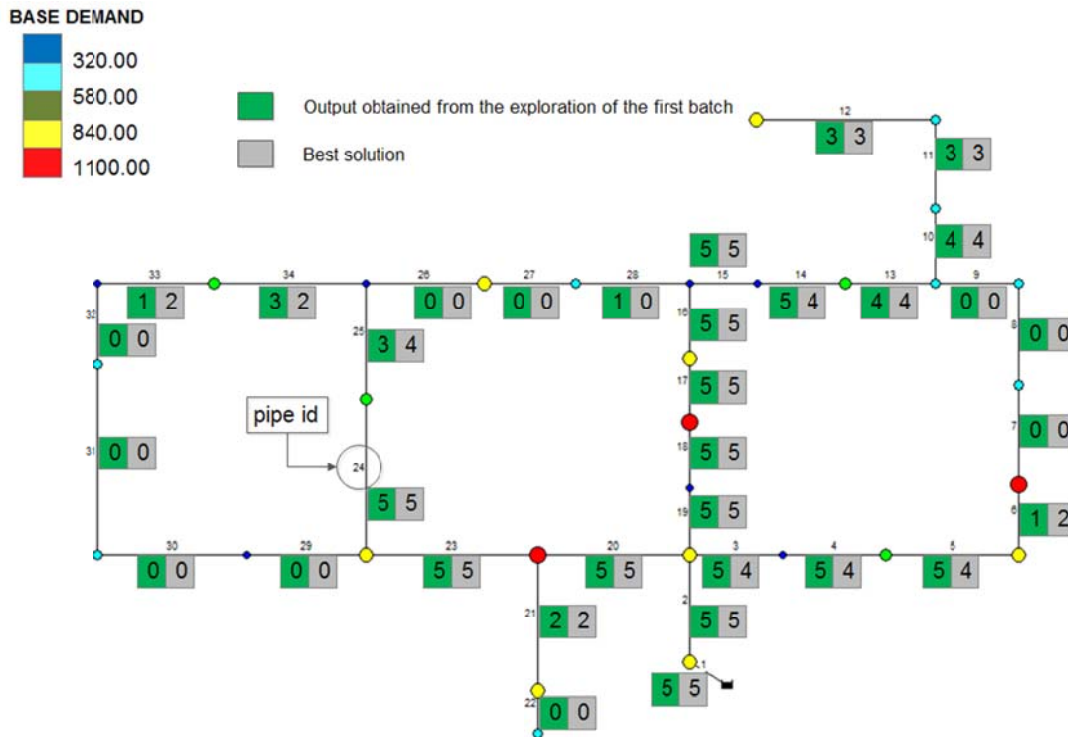


Figura 3. Red de Hanoi con *codebooks* para el primer y segundo SOM (este último, también mejor solución).

El *codebook* correspondiente a esta neurona se utiliza para asignar los valores correspondientes a los tubos – considerando adicionalmente una pequeña cantidad de aleatoriedad en lugar de aceptar los valores del *codebook* como reglas duras para ser aplicadas.

Después de la aplicación de estas reglas, los dos *swarms* continuaron con la iteración. En la ejecución específica que estamos describiendo, la convergencia para el *swarm* que utilizó el

conocimiento del SOM se produjo en la iteración 160 (la condición de terminación permite 500 iteraciones más, sin mejora). El óptimo correspondió a una red con un coste de 6.545325 millones de dólares y diámetros de tuberías como se indica en la Figura 3 (cuadrados de la derecha en cada par). Una nueva base de datos se fue generando mientras que el segundo *swarm* seguía iterando. Mediante el uso de la nueva base de datos recogidos durante esta última parte del proceso de iteración, se obtuvo un nuevo SOM. Este mapa se representa en la Figura 4. Señalemos que el *codebook* que agrupa la mayor parte de las buenas soluciones coincide completamente con el vector de diámetros asociados con la mejor solución.

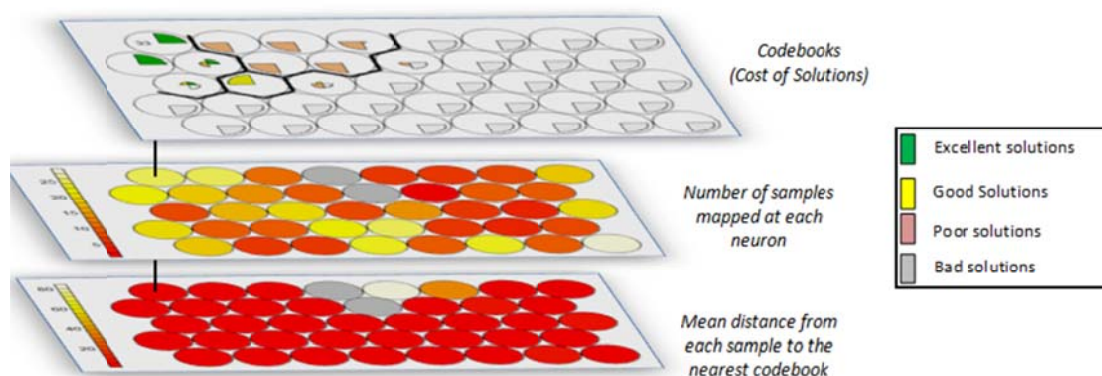


Figura 4. SOM que recoge los pasos de las iteraciones finales

Conviene mencionar un segundo aspecto: la facilidad con la que un SOM puede ser generado - incluso para una base de datos relativamente grande. Los SOM presentados en las Figuras 2 y 4 convergieron en aproximadamente 15 segundos usando un ordenador Intel (R) core con 2,70 GB de RAM utilizable. Como consecuencia, la obtención de áreas de mayor probabilidad a través de SOMs es mucho menos costosa que el funcionamiento de las simulaciones de redes hidráulicas que habrían sido necesarios si no se hubiera utilizado el SOM. Podemos concluir que pueden generarse fácilmente áreas de búsqueda de probabilidad que no han podido ser evaluadas utilizando el conocimiento experto. Su evaluación sólo se basa en los datos y la estructura de los patrones encontrados. Sugerimos que, además del conocimiento experto, estas reglas pueden ser de gran interés para ayudar a un AE a restringir la búsqueda a áreas más prometedoras, lo que reduce el tamaño del espacio de búsqueda.

Un tercer punto interesante es el siguiente. Al comparar los SOM de las Figuras 2 y 4 se observa fácilmente las diferentes distribuciones de sus diversos tipos de soluciones. En el SOM en la Figura 2, después de la primera parada de iteración, la diversidad de las soluciones actuales es baja, como puede verse por el número relativamente grande de neuronas que han capturado excelentes y buenas soluciones. Por el contrario, en el SOM en la figura 4, que representa el retrato final de la historia de la evolución, la diversidad ha aumentado, y, como resultado, las soluciones buenas y excelentes son relativamente escasas y se concentran en menos neuronas.

Para proporcionar un apoyo adicional para el enfoque que aquí se presenta, en la Figura 5 se

presentan las curvas de la evolución de los costes con la iteración. Se puede observar cómo, después de la inyección de conocimiento obtenido con el SOM construido después de la primera tanda de iteración, el *swarm* que ha utilizado tal conocimiento reduce rápidamente el coste, convergiendo, por lo tanto, más rápido.

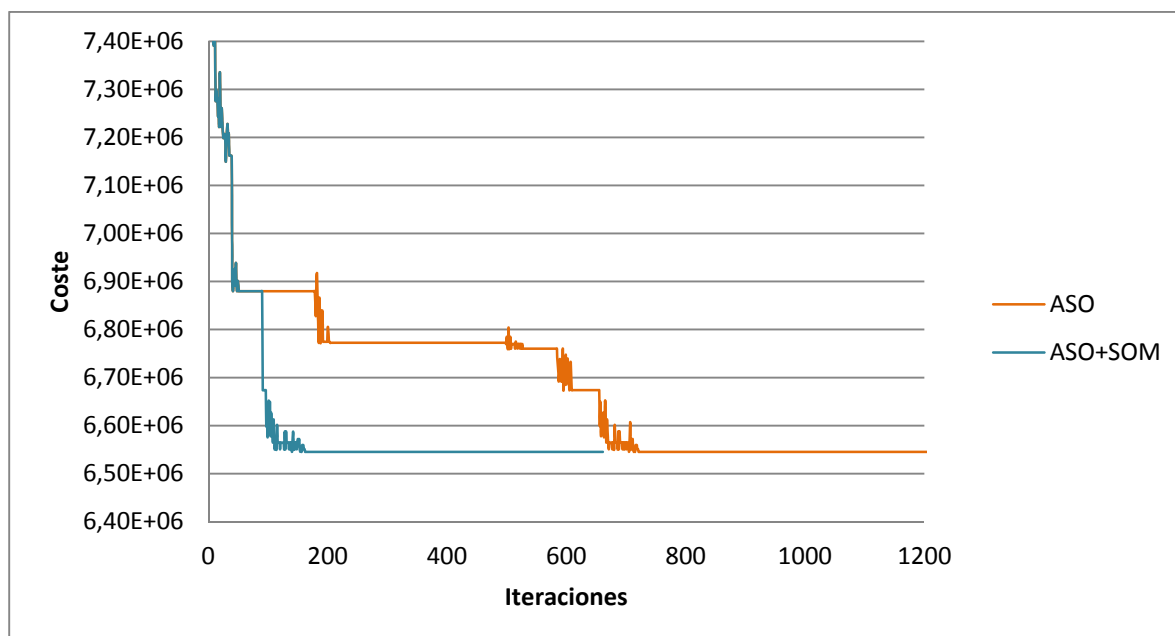


Figura 5. Históricos de la evolución con y sin SOM para el problema de Hanoi

Podemos concluir que se puede generar conocimiento que no se ha obtenido con el conocimiento experto. La evaluación se basa únicamente en los datos y las estructuras de los patrones encontrados. Sostenemos que, además del conocimiento de los expertos, este conocimiento ‘sintético’ puede ser de gran interés para ayudar al proceso de optimización evolutiva restringiendo la búsqueda a las áreas más prometedoras, lo que reduce el tamaño del espacio de búsqueda.

5. CONCLUSIONES

En este trabajo hemos presentado un enfoque basado en la optimización distribuida para resolver problemas de optimización realistas. El enfoque propuesto es una estructura que integra varias metaheurísticas con características diferentes que cooperan. Los parámetros utilizados por las distintos metaheurísticas se ajustan de forma adaptativa y auto-adaptativa, utilizando los mecanismos de optimización incorporados en cada metaheurística. Los agentes se basan en un sistema basado en el conocimiento que encapsulan tanto reglas dependientes del problema, como otras reglas obtenidas mediante la integración de elementos de las tecnologías de inteligencia artificial como DM. En concreto, hemos considerado mapas SOM. Para administrar la complejidad asociada a la solución de problemas, hemos utilizado un MAS, en el que se ha desarrollado una plataforma de simulación y experimentación

distribuida. Esta plataforma proporciona la infraestructura de uso general adecuado para la optimización numérica utilizando diversas categorías de heurísticas.

La interacción que ofrece el intercambio de información y los mecanismos de adaptación proporcionan un elemento clave para el desarrollo de efectos emergentes y sinérgicos que conducen a soluciones más eficaces para una amplia variedad de problemas de optimización realistas.

Varias ventajas del enfoque de optimización propuesto pueden, además, ser mencionadas. En primer lugar, los resultados de la búsqueda se reutilizan y selectivamente se almacenan en una base de datos dinámica de la que se extraen reglas o zonas de mayor probabilidad con el objetivo de mejorar la eficiencia de búsqueda. En segundo lugar, el entorno definido es escalable con respecto a su capacidad de expansión por otros métodos de optimización o tecnologías encapsuladas en agentes. En tercer lugar, la modelación de la complejidad del enfoque de optimización híbrido es relativamente bajo, debido al hecho de que no es el comportamiento global del entorno lo que tiene que ser definido de forma explícita, sino sólo el comportamiento local de los agentes. Como resultado, y en cuarto lugar, el enfoque es directamente aplicable a otros problemas de ingeniería. Por último, pero no menos importante, la interacción de uno o más usuarios con la plataforma proporciona una mejor usabilidad del marco computacional. Por el contrario, el principal inconveniente de este enfoque es la creciente complejidad de diseño e implementación de software para la optimización basada en agentes.

Teniendo en cuenta la tecnología existente desde los puntos de vista de software y hardware, la capacidad para resolver problemas complejos de optimización se ha incrementado significativamente en comparación con la situación de hace diez años. En el campo del agua, como en otros problemas de ingeniería, el mayor reto hoy en día sigue siendo menos la capacidad de resolver un problema y mucho más la capacidad de identificar el planteamiento del problema adecuado correspondiente a las necesidades reales de la situación que hay que resolver. Desde hace varios años, los planteamientos de los problemas se han visto limitados por las técnicas disponibles para resolverlos. Hoy en día, técnicas como la presentada en esta investigación, abren una puerta a un mundo de muchas posibilidades. En el caso del diseño de sistemas de distribución de agua, por ejemplo, las buenas soluciones de ingeniería no vendrán de un superalgoritmo de optimización, sino de un planteamiento del problema adecuado y un algoritmo capaz de reproducir el pensamiento de ingeniería.

REFERENCIAS

- [1] J.W. Nicklow, P.M. Reed, D. Savic, T. Dessalegne, L. Harrell, A. Chan-Hilton, M. Karamouz, B. Minsker, A. Ostfeld, A. Singh, y E. Zechman, “State of the Art for Genetic Algorithms and Beyond in Water Resources Planning and Management”, *J. Water Resour. Plan. Manag. ASCE* Vol. 136 (4), pp. 412-432, (2010).
- [2] V.V. Nguyen, D. Hartmann y M. König, “A distributed agent-based approach for simulation-based optimization”, *Adv. Eng. Informat.* Vol. 26, pp. 814–832, (2012).
- [3] P.M. Reed, D. Hadka, J.D. Herman, J.R. Kasprzyk y J.B. Kollat, “Evolutionary multiobjective optimization in water resources: The past, present, and future”, *Advan.*

- Water Resour.* Vol. 51, pp. 438-456, (2013).
- [4] A.C. Zecchin, A.R. Simpson, H.R. Maier, A. Marchi, y J.B. Nixon, “Improved understanding of the searching behavior of ant colony optimization algorithms applied to the water distribution design problem”. *Water Resour. Res.* Vol. 48 (9), W09505, (2012).
 - [5] A. Marchi, G. Dandy, A. Wilkins y H. Rohrlach, “Methodology for comparing evolutionary algorithms for optimization of water distribution systems”, *J. Water Resour. Plan. Manag.* Vol.140 (1), pp. 22-31, (2014).
 - [6] S.Y. Liong y M. Atiquzzama, “Optimal design of water distribution network using shuffled complex evolution”. *J. Inst. Eng. Singapore* Vol. 44 (1), pp. 93-107, (2004).
 - [7] Z.W. Geem, “Optimal cost design of water distribution networks using harmony search”. *Eng. Optim.* Vol. 38 (3), pp. 259-280, (2006).
 - [8] J. Izquierdo, I. Montalvo, R. Pérez y M. Tavera, *Optimization in water systems: a PSO approach*. Business and Industry Symposium (BIS), Ottawa, Canada, (2008).
 - [9] X. Jin, J. Zhang, J.L. Gao y W.Y. Wu, “Multi-objective optimization of water supply network rehabilitation with non-dominated sorting Genetic Algorithm-II”. *J. Zhejiang Univ. SCIENCE A* Vol. 9 (3), pp. 391-400, (2008).
 - [10] I. Montalvo, J. Izquierdo, S. Schwarze y R. Pérez-García, “Multi-objective particle swarm optimization applied to water distribution systems design: An approach with human interaction”. *Math. Comput. Model.* Vol. 52, pp. 1219-1227, (2010).
 - [11] H. Shean y E. McBean, *Hydraulic calibration for a small water distribution network*. K. Lansey et al. eds. *Proc. 12th Water Distribution Systems Analysis Symp, Tucson, Arizona*, (2010).
 - [12] W. Bei y G.C. Dandy, *Retraining of metamodels for the optimization of water distribution systems*, *The 14th Water Distribution Systems Analysis Conference*, ASCE, Adelaide, South Australia., pp. 36-47, (2012).
 - [13] L. Berardi, D. Laucelli y O. Giustolisi, *A decision support tool for operational optimization in WDNEXL*, *The 14th Water Distribution Systems Analysis Conference*, ASCE, Adelaide, South Australia, pp. 48-65, (2012).
 - [14] Z.Y. Wu y M. Behandish, *Real-time pump scheduling using genetic algorithm and artificial neural network based on graphics processing unit*. *Proc. Water Distribution System Analysis Conference*, Adelaide, Australia, 2012, pp. 1088-1099, (2012).
 - [15] I. Montalvo, J. Izquierdo, M. Herrera R. y Pérez-García, “Water supply system computer-aided design by Agent Swarm Optimization”, *Comput. Aided Civil Infrast. Eng.* Vol. 29 (6), pp. 433-448, (2014).
 - [16] I. Montalvo, *Diseño óptimo de sistemas de distribución de agua mediante Agent Swarm Optimization*. PhD doctoral dissertation. Universitat Politècnica de València, Valencia, Spain, (2011).
 - [17] K.P. Sycara, “Multiagent systems, American Association for Artificial Intelligence”, *AI Mag.* Vol. 19 (2), pp. 79-92, (1998).
 - [18] M. Wooldridge, *An Introduction to Multiagent Systems*, John Wiley & Sons, (2009).
 - [19] G. Weichhart, M. Affenzeller, A. Reitbauer y S. Wagner, *Modelling of an agentbased schedule optimisation system*, *Proceedings of the IMS International Forum*, (2004).

- [20] J.A. Persson, P. Davidsson, S.J. Johansson y F. Wernstedt, *Combining agent-based approaches and classical optimization techniques*, *Proc. of the European workshop on Multi-Agent Systems (EUMAS 2005)*, pp. 260–269, (2005).
- [21] X.F. Xie y J. Liu, “Graph coloring by multiagent fusion search”. *Combinat. Optim.* Vol. 18 (2), pp. 99–123, (2009).
- [22] J. Izquierdo, I. Montalvo, R. Pérez y V.S. Fuertes, “Design optimization of wastewater collection networks by PSO”, *Comput. Math. Appl.* Vol. 56 (3), pp. 777–784, (2007).
- [23] J. Izquierdo, R. Minciardi, I. Montalvo, M. Robba y M. Tavera, *Particle Swarm Optimization for the biomass supply chain strategic planning*, *Proceedings of 4th Biennial Meeting, iEMSs 2008: International Congress on Environmental Modelling and Software*, 1272-1280, Barcelona, Spain, (2008).
- [24] J.L. Díaz, M. Herrera, J. Izquierdo, I. Montalvo y R. Pérez-García, *A Particle Swarm Optimization derivative applied to cluster analysis*, *Proceedings of 4th Biennial Meeting, iEMSs 2008: International Congress on Environmental Modelling and Software*, Barcelona, Spain, (2008).
- [25] E. Campbell, J. Izquierdo, I. Montalvo, A. Ilaya-Ayza, R. Pérez-García y M. Tavera, “A Flexible Methodology to Sectorize Water Supply Networks Based on Social Network Theory Concepts and on Multi-objective Optimization”, *J. Hydroinf.*, to appear, (2015).
- [26] M. Clerc, *Particle Swarm Optimization*. ISTE Ltd., (2006).
- [27] S. Lessmann, M. Caserta e I. Montalvo, “Tuning metaheuristics: A data mining based approach for particle swarm optimization”, *Expert Systems with Applications: An international Journal*, Vol. 38 (10), pp. 12826-12838, (2011).
- [28] J.A. Vrugt y B.A. Robinson, *Improved evolutionary optimization from genetically adaptive multimethod search*, *Proc. Natl. Acad. Sci. U S A.* 2007 January 16; 104(3): 708–711, (2007).
- [29] J.A. Vrugt, B.A. Robinson y J.M. Hyman, “Self-adaptive multimethod search for global optimization in real-parameter spaces”, *IEEE Trans. Evol. Comput.* Vol. 13 (2), pp. 243-259, (2009).
- [30] D. Hadka y P. Reed, “Borg: An auto-adaptive many-objective evolutionary computing framework”. *Evol. Comput.* Vol. 2012, pp. 1-30, (2012).
- [31] K. McClymont, E.C. Keedwell, D. Savic y M. Randall-Smith, “A General Multi-objective Hyper-Heuristic for Water Distribution Network Design with Discolouration Risk”, *J. Hydroinform.* Vol. 5 (3), pp. 700-716, (2013).
- [32] I. Montalvo, J. Izquierdo, R. Pérez y M. Herrera, “Improved performance of PSO with self-adaptive parameters for computing the optimal design of Water Supply Systems”, *Eng. Appl. Artif. Intel.* Vol. 23 (5), pp. 727-735, (2010).
- [33] T.G. Crainic, Y. Li y M. Toulouse, “A first multilevel cooperative algorithm for capacitated multicommodity network design”, *Comput. Operat. Res.* Vol. 33(9), pp. 2602–2622, (2006).
- [34] T.G. Crainic y M. Toulouse, *Explicit and emergent cooperation schemes for search algorithms*, *Learning and Intelligent Optimization Conference*, Vol. 5315 of *Lecture Notes in Computer Science*, pp. 95–109, (2008).

- [35] A. Nedic y A. Ozdaglar, “Distributed subgradient methods for multi-agent optimization”, *IEEE Trans. Autom. Control* Vol. 54(1), pp. 48–61, (2007).
- [36] E.G. Talbi y V. Bachelet, *Cosearch: a parallel cooperative metaheuristic*, C. Blum, A. Roli, M. Sampels eds., *First International Workshop on Hybrid Metaheuristics* (HM 2004), pp. 127–140, (2004).
- [37] E.G. Talbi, S. Mostaghim, T. Okabe, H. Ishibuchi, G. Rudolph y C. Coello-Coello, *Parallel approaches for multiobjective optimization*, Jrgen Branke et al. eds., *Multiobjective Optimization, Lecture Notes in Computer Science*, Vol. 5252, Springer, Berlin/Heidelberg, pp. 349–372, (2008).
- [38] T. Kohonen, *Self-Organizing Maps*, Springer-Verlag, Berlin, Heidelberg, (2001).
- [39] T. Kohonen, “Essentials of the self-organizing map”. *Neural Netw.* Vol. 37, pp. 52–65, (2013).
- [40] M.A. Kraaijveld, J. Mao y A.K. Jain, “A nonlinear projection method based on Kohonen's topology preserving maps”. *IEEE Trans. Neural Netw.* Vol. 6 (3), pp. 548–559, (1995).
- [41] K. Tasdemir y E. Merenyi, “Exploiting Data Topology in Visualization and Clustering of Self-Organizing Maps”. *IEEE Trans. Neural Netw.* Vol. 20 (4), pp. 549–562, (2009).
- [42] W. Melsen, R. Wehrens y L. Buydens, “Supervised Kohonen networks for classification problems”. *Chemom. Intell. Lab. Syst* Vol. 83, pp. 99–113, (2006).
- [43] R. Wehrens y L.M.C. Buydens, “Self- and Super-organizing Maps in R: The Kohonen Package”. *J. Stat. Softw.* Vol. 21 (5), pp. 1–19, (2007).
- [44] M.C. Cunha y J. Sousa, “Water distribution network design optimization: simulated annealing approach”. *J. Water Resour. Plan. Manag.* Vol. 125 (4), pp. 215–221, (1999).
- [45] A.S. Matías, *Diseño de redes de distribución de agua contemplando la fiabilidad, mediante Algoritmos Genéticos*. PhD doctoral dissertation. Universitat Politècnica de València, Valencia, Spain, (2003).
- [46] D.A. Savic y G.A. Walters, *Genetic operators and constraint handling for pipe network optimization*. *Evolutionary Computing*, AISB Workshop 1995, 154–165, (1995).
- [47] A.C. Zecchin, A.R. Simpson, H.R. Maier y J.B. Nixon, “Parametric study for an ant algorithm applied to water distribution system optimization”. *IEEE Trans. Evol. Comput.* Vol. 9 (2), pp. 175–191, (2005).
- [48] I. Montalvo, J. Izquierdo, R. Pérez y M.M. Tung, “Particle Swarm Optimization applied to the design of water supply systems”, *Comput Math. Appl.* Vol. 56(3), pp. 769–776, (2008).
- [49] I. Montalvo, J. Izquierdo, R. Pérez y P.L. Iglesias, “A diversity-enriched variant of discrete PSO applied to the design of Water Distribution Networks”. *Eng. Optim.* Vol. 40(7), pp. 655–668, (2008).
- [50] Z.Y. Wu y A.R. Simpson, “Competent genetic-evolutionary optimization of water distribution systems”. *J. Comput. Civil Eng.* Vol. 15 (2), pp. 89–101, (2001).