Algoritmos Basados en Inteligencia Colectiva para la Resolución de Problemas de Bioinformática y Telecomunicaciones

Trabajo de Investigación Tutelado

Programa de Máster en Ingeniería del

Software e Inteligencia Artificial

Periodo 2006/2007

Alumno: José Manuel García Nieto
Tutor: Enrique Alba Torres

Universidad de Málaga, Septiembre de 2007

Resumen

En este trabajo se estudia la resolución de problemas procedentes del dominio de la Bioinformática y las Telecomunicaciones mediante algoritmos bioinspirados. En concreto, nos centramos en una subclase de éstos basada en procesos de inteligencia colectiva (*Swarm Intelligence*). Actualmente, existe un creciente interés en los problemas relacionados con el descubrimiento de propiedades en el campo de la genética, siendo especialmente sensible en los estudios sobre análisis de cánceres. Un foco diferente de interés en la investigación científica, aunque en cierto sentido relacionado con el anterior, reside en el campo de las telecomunicaciones, debido a la gran demanda de servicios de comunicación por parte de la sociedad. Muchos de estos problemas pueden plantearse como problemas de optimización que no pueden resolverse de forma exacta o exhaustiva debido al gran tamaño de su espacio de soluciones. En tales casos, las técnicas metaheurísticas se convierten en una buena opción para encontrar alguna solución. Estas técnicas no aseguran una solución óptima pero en la mayoría de los casos pueden obtener una solución aceptablemente buena en un tiempo razonable. Dentro de las metaheurísticas se encuentran los algoritmos bioinspirados, que se inspiran en algunos procesos naturales emulando el "*modus operandi*" de ciertos animales cuando se enfrentan a problemas de supervivencia. Tras una breve revisión sobre problemas de bioinformática y telecomunicaciones, se realiza una visión global de los algoritmos de inteligencia colectiva. A continuación, el trabajo aborda la resolución del problema de la selección y clasificación de genes en Microarrays de ADN (como parte bioinformática) y el problema de la gestión de la localización de terminales móviles en redes GSM (como parte de telecomunicaciones).

Índice general

Capítulo 1

Introducción

Uno de los principales frentes de trabajo en el ámbito de la Informática ha sido tradicionalmente el diseño de algoritmos cada vez más eficientes para la solución de problemas tanto de optimización como de búsqueda. En este dominio, el objetivo consiste en obtener algoritmos nuevos que necesiten un esfuerzo computacional más pequeño que los algoritmos existentes con la intención de abordar los problemas de forma más eficiente y también para poder abordar tareas que estaban vedadas debido al coste de otras técnicas clásicas.

La investigación en algoritmos tanto exactos como heurísticos para resolver problemas de optimización combinatoria tiene una vigencia inusualmente importante en estos días, ya que nos enfrentamos a nuevos problemas de ingeniería al mismo tiempo que contamos con nuevos recursos computaciones tales como máquinas más potentes, redes y entornos como Internet.

Los algoritmos basados en inteligencia colectiva o *Swarm Intelligence* (SI), es un novedoso paradigma de inteligencia distribuida para la resolución de problemas de optimización, que toma su inspiración en ejemplos biológicos de comportamiento colectivo (*swarming*) como el fenómeno de las bandadas y las manadas en los vertebrados. Los algoritmos de cúmulos de partículas o *Particle Swarm Optimization* (PSO) [26] incorporan comportamientos colectivos observados en las bandadas de aves, los bancos de peces, las colmenas de abejas e incluso en el comportamiento social de los humanos. Los algoritmos de colonias de hormigas, conocidos como Ant Colony Optimization (ACO) [13], trabajan con sistemas artificiales que se inspiran en el comportamiento de búsqueda de las hormigas reales, siendo utilizados para resolver problemas de optimización discreta.

Este tipo de técnicas metaheurísticas bioinspiradas vienen siendo aplicadas con éxito durante esta última década y actualmente en multitud de problemas de optimización, ya sea en problemas de naturaleza discreta, continua e incluso con representación de árboles y grafos. La idea de reutilizar mecanismos que *siempre han existido en la naturaleza* para su aplicación en problemas reales, introduce un componente de interés público en otras disciplinas de la ciencia cómo la biología, psicología e incluso la reciente nanotecnología. Supone un campo totalmente abierto a la investigación, por lo que suscita un gran interés por parte de la comunidad científica.

Dentro de los dominios de conocimiento que pueden beneficiarse de estos eficientes algoritmos de optimización, la investigación microbiológica a nivel molecular está viviendo un cambio espectacular en los últimos años. En apenas una década se ha pasado de trabajos basados en el estudio de uno o unos pocos genes al análisis de los genomas en su totalidad. La técnica de Microarrays de ADN (MAs) [4] está atrayendo un gran interés, ya que permite monitorizar la actividad de un genoma completo mediante un simple experimento. Además, cada uno de estos experimentos con MAs puede suponer el manejo desde cientos hasta decenas de miles de genes, normalmente con decenas de muestras por gen. Por este motivo, son necesarias técnicas de reducción que permitan agrupar genes con patrones de expresión relacionados e informativos, desechando la información irrelevante. Algunos ejemplos son las técnicas de clustering [9], reconocimiento de patrones o reglas [23], selección y clasificación de genes en MAs [15].

Por otra parte, las telecomunicaciones son un símbolo importante de nuestra actual sociedad de la información. Con un rápido crecimiento de servicios para el usuario, la telecomunicación es un campo en el que hay muchas líneas de investigación abiertas que están desafiando a la comunidad investigadora. Muchos de los problemas que podemos encontrar en esta área pueden ser formulados como problemas de optimización. Algunos ejemplos son: la signación frecuencias en enlaces de radio [33], la gestión de localización de terminales móviles en redes GSM [48] y diseñar la red de telecomunicaciones [7].

En la práctica, la mayoría de estos problemas de optimización son inabordables con técnicas exactas. Por eso, es adecuado el uso de enfoques heurísticos tales como los algoritmos evolutivos y bioinspirados. No obstante, la gran complejidad de las instancias reales de esos problemas requieren la potencia de cálculo de varias máquinas trabajando juntas para encontrar soluciones aceptables. Esto da lugar a la aplicación de algoritmos paralelos para abordar eficientemente esos problemas.

El presente trabajo se ha estructurado de la siguiente manera. En primer lugar se realiza un estudio de diferentes problemas del dominio de la Bioinformática y las Telecomunicaciones en el Capítulo 2. De entre ellos se ha escogido dos con aplicación real directa para resolverlos con algoritmos de inteligencia colectiva: la selección y clasificación de genes en Microarrays de ADN, como ejemplo de bioinformática y la gestión de la localización óptima de terminales móviles en redes GSM, como exponente de las telecomunicaciones. Posteriormente, en el Capítulo 3 se hace una visión global de de los algoritmos de Swarm Intellegence, describiéndose de forma general y centrándose especialmente en los algoritmos usados en la tercera parte del trabajo. En ésta (Capítulos 4 y 5), se detallan los problemas abordados y se presentan los experimentos y los resultados obtenidos. Por último, se concluye en el Capítulo 6 con las principales reflexiones obtenidas de este trabajo por parte del alumno.

Capítulo 2

Problemas de Bioinformática y Telecomunicaciones

Este capítulo desarrolla una descripción de los problemas abordados en este trabajo pertenecientes al campo de la bioinformática y las telecomunicaciones. Como se ha mencionado anteriormente, se han elegido los problemas de "Selección y Clasificación de Genes en Microarrays de ADN" y la "Gestión de la Localización Óptima de Terminales Móviles en Redes GSM", ambos de complejidad NP-difícil. En primer lugar se realiza un breve repaso por los artículos más relacionados del estado del arte, para pasar a la motivación y definición teórica de cada problema.

2.1. Selección y Clasificación de Genes en Microarrays de ADN

Para comenzar, en la Tabla 2.1 mostramos algunos de los últimos trabajos publicados donde se resuelve el problema de la selección y clasificación de genes en Microarrays de niveles de expresión de ADN con técnicas metaheurísticas. Además, se muestra el algoritmo clasificador empleado.

Tabla 2.1: Resumen de artículos que resuelven la selección y clasificación de genes en Microarrays de niveles de expresión de ADN

Autor(es)	Año	Algoritmo	Clasificador
Li et al. [29]	2001	Genetic Algorithm	k-Nearest Neighbor
Guyon et al. [20]	2002	Recursive Feature Elimination	Support Vector Machines
Deb and Reddy [11]	2003	Multiobjective EA (NSGA-II)	k-Nearest Neighbor
Liu et al. [31]	2004	Neural Network	k-Nearest Neighbor
Yu and Liu [52]	2004	Two-Step Redundancy Based Filter	Correnation
Juliusdottir et al. [24]	2005	Two Phase Evolutionary Algorithm	k-Nearest Neighbor
Shen Qi et al. [39]	2006	Particle Swarm Optimization	Support Vector Machines
Banerjee et al. [3]	2007	MOGA with Rough Sets	k-Nearest Neighbor
Alba et al. [15]	2007	Geometric PSO	Support Vector Machines

Mediante la tecnología de Microarrays de ADN [37] es posible analizar decenas de miles de genes en un simple experimento y de este modo extraer información significativa de la función celular, ya que los cambios en la fisiología de un organismo están generalmente asociados con cambios en los patrones de lo niveles de expresión de sus genes. Existen MAs obtenidos de tumores reales como *AML ALL Leukemia* [18], *Colon* [2] y *Breast* [49] los cuales están siendo estudiados y comparados con niveles de expresión de muestras sanas. Sin embargo, en estos experimentos existe una gran cantidad de información redundante y errónea, de hecho, la gran mayoría de las muestras obtenidas se creen no informativas con respecto a las clases estudiadas y únicamente una pequeña fracción de los genes presentan perfiles diferenciadores y significativos. Por este motivo, el uso de técnicas para tratar con estos datos es sumamente necesario. Estas técnicas deben aprender a identificar subconjuntos de genes informativos enmarañados en enormes bases de datos "contaminadas" con información redundante [24].

En este contexto, la selección de características (*feature selection*) se considera como una herramienta de preproceso necesaria para analizar este tipo de datos, ya que este método puede reducir la dimensión de las bases de datos y conducir a mejores análisis [20]. Feature selection utiliza métodos wrapper [28] para la clasificación, con el objetivo de discriminar cierto tipo de tumor, para reducir el número de genes a investigar en el caso de un nuevo paciente, además de asesorar en el descubrimiento de medicamentos y el diagnóstico temprano. Como técnicas de clasificación son comúnmente utilizados, y así se muestra en la tabla anterior, el algoritmo K-Nearest Neighbor (K-NN) [16] y Support Vector Machines (SVM) [8] (utilizado en este trabajo). Mediante la creación de clústeres, se obtiene una gran reducción del número de genes manteniendo la calidad de la clasificación (*accuracy*).

El problema de feature selection se define como: dado un conjunto de características (genes) $F = \{f_1, ..., f_i, ..., f_n\}$, encontrar un subconjunto $F' \subseteq F$ que maximize una función de evaluación $\Theta : \Gamma \rightarrow G$ tal que

$$F' = argmax_{G \subset \Gamma}\{\Theta(G)\}, \tag{2.1}$$

donde Γ es el espacio de todos los posibles subconjuntos de F y G sea un subconjunto de Γ. Este problema ha sido catalogado como NP-dificil [36]. Por lo tanto, para tratar con este problema se hace imprescindible el uso de técnicas metaheurísticas. Recientemente, algunos ejemplos de éstas como los GAs y PSO han sido utilizados en este contexto [51, 24, 21].

En este trabajo, se propone la utilización de dos modelos híbridos basados en algoritmos bioinspirados con algoritmos puros de clasificación. Principalmente, se trata el uso un algoritmo basado en cúmulos de partículas (PSO) [27] (*Swarm Intelligence*) combinado con máquinas de vectores de soporte (SVM). Concretamente, se utiliza una versión novedosa de PSO llamada *Geometric PSO* [34] que es explicada en el siguiente capítulo (Section 3.2). Como modelo adicional se utiliza un GA Generacional el cual utiliza un operador de cruce especializado: *Size-Oriented Common Feature Crossover Operator* (SSOCF) [22], también combinado con SVM.

Ambos algoritmos son evaluados utilizando seis conocidas bases de datos de cánceres reales (Leukemia, Colon, Breast, Ovarian [38], Prostate [41] y Lung [19]), comparando sus prestaciones y la calidad de las soluciones obtenidas.

2.2. Gestión de la Localización en Redes GSM

En esta sección se presenta el problema de la gestión de la localización en redes GSM, conocido en la literatura como *Location Area Management* (LA). Se aborda en este trabajo mediante algoritmos de cúmulos e inteligencia colectiva, elegido como ejemplo del campo de las telecomunicaciones. Del mimo modo al problema anterior, en la Tabla 2.2 se citan algunos artículos del estado del arte donde se trata este problema.

Tabla 2.2: Resumen de artículos más relevantes en los que se aborda el problema de Location Area Management mediante técnicas metaheurísticas

Autor(es)	Año	Algoitmo
Demirkol et al. [12]	2001	Simulated Annealing
Subrata and Zomaya [43]	2003	Cellular Automata
Subrata and Zomaya [42]	2003	Genetic Algorithm Ant Colony Optimization Tabu Search
Taheri and Zomaya [44]	2004	Hopfield Neural Network
Taheri and Zomaya [46]	2005	Genetic Algorithm
Taheri and Zomaya [47]	2005	Simulated Annealing

En la actualidad, numerosas empresas y proveedores de servicios demandan soluciones completamente integrados de redes de móviles inalámbricos. Los servicios de voz IP, fax y servicios de pago serán combinados con transferencia de datos, video-conferencia y otros servicios multimedia para hacer la próxima generación de terminales móviles inalámbricos, como las redes IMT-2000 y UMTS, más atractivos. Básicamente, estas redes han sido diseñadas para soportar una verdadera combinación de servicios de tiempo real para formar una red global personal de comunicaciones [30, 1]. Con el objetivo de soportar tal rango de transferencia de datos y de aplicaciones de usuarios, la gestión de la movilidad de los terminales en la red comienza a ser una cuestión crucial en el diseño de infraestructuras para redes móviles inalámbricas. La Figura 2.1a muestra ejemplos de configuraciones de celdas de redes GSM.

Las peticiones de gestión de movilidad son de esta forma inicializadas bien por el movimiento de un terminal cuando cruza la frontera de una celda de cobertura, o bien, por el deterioro en la calidad de la señal recibida en un canal localizado. Debido al esperado incremento en el uso de los servicios inalámbricos (wireless) en un futuro cercano, la nueva generación de redes de móviles deben ser capaces de soportar un gran número de usuarios y gestionar sus necesidades de ancho de banda.

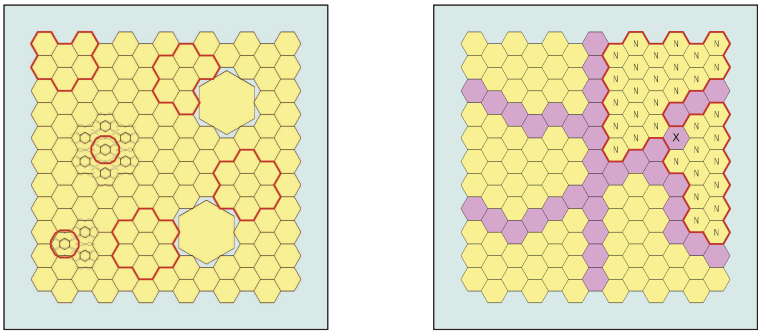


Figura 2.1: (a) Ejemplo de red GSM y (b) configuración de celdas de paging

En las redes GSM, las celdas son agrupadas juntas en regiones con un cierto espectro de frecuencia asignado. Sin embargo, las técnicas de *paging* intentan manejar este problema desde un punto de vista completamente diferente. Básicamente, la gestión de la localización consiste en dos funciones principales: la *actualización de la posición* y la *búsqueda de la posición*. En la primera, todos los terminales móviles actualizan sus localizaciones en la red y notifican a la propia red su situación actual. Por otra parte, en la búsqueda, es la red la que realiza el esfuerzo de localizar a un usuario basándose en la última localización.

Es esquema de *Paging Cells* (PCs) es uno de los métodos más populares utilizados para la reducción del coste total del las operaciones de gestión de la localización de terminales. Según esta estrategia, un usuario realiza la actualización cuando cruza ciertas celdas predefinidas conocidas como celdas de paging [50] [43]. En tal caso, se define un vecindario de paging para cada celda. El vecindario de cada celda de paging contiene todas las celdas que no son de paging (que no realizan gestión) que deben llamar al usuario en el caso de recibir una llamada entrante. Las celdas de paging se muestran en gris en la Figura 2.1b, donde las celdas vecinas de cada celda de paging 'X' se marcan con 'N' formando el vecindario de dicha celda.

En la Tabla 2.2 se resumen los trabajos principales en los que se aborda el problema y los algoritmos utilizados. En todos estos se define el problema del mismo modo, es decir, gestionando el coste de la gestión de la localización mediante dos partes principales: el coste de actualización y el coste de llamada (paging). El coste de actualización es una porción del coste total debido a los movimientos realizados por los terminales móviles en su deambular por la red, mientras que el coste de llamada es causado por la red durante una búsqueda cuando se intenta localizar a un usuario. En realidad, el coste total incluye otros costes como el de gestión de la base de datos de usuarios en los registros, los costes de la red cableada (dorsal) cuando conecta las estaciones base entre ellas, el coste de enrutado en las estaciones base en el caso de desvío de llamadas y otros componentes.

No obstante, estos costes adicionales son iguales para todas las estrategias de red, así que no son considerados en el problema. Como resultado de la combinación de los dos costes tomados en consideración, se define el coste total como:

$$Cost = \beta \times N_{LU} + N_P \tag{2.2}$$

donde N_{LU} es el número total de actualizaciones, N_P representa el número total de transacciones de paging y β es una constante que representa la proporción de coste de una actualización respecto a una operación de paging en la red. Esto se debe a que empíricamente, el coste de cada actualización es mucho mayor al de una transacción de paging (10 veces más en la mayoría de los casos) [50] [42] [45].

En el Capítulo 5, se describe la utilización de un algoritmo de cúmulo de partículas en la resolución del problema de la gestión de localización de terminales móviles tal y como se ha introducido aquí. En concreto se trata del algoritmo Geometric PSO, sobre el cual se han realizado experimentos de evaluación con doce instancias de redes de celdas GSM.

Capítulo 3

Algoritmos Basados en Inteligencia Colectiva

En este capítulo se realiza una breve introducción a las técnicas metaheurísticas bioinspiradas y en especial de los algoritmos de inteligencia colectiva. Como eje central de este trabajo, se hace una descripción más detallada del algoritmo de cúmulos basado en "espacios geométricos", que será aplicado a la resolución de los problemas presentados en el capítulo anterior (secciones 2.1 y 2.2). Así pues, en la siguiente sección se hará un recorrido a vista de pájaro por las técnicas de Swarm Intelligence más representativas, para centrarse en la última sección en los algoritmos empleados en los experimentos de la tercera parte del trabajo.

3.1. Swarm Intelligence: Una Visión General

En un problema de optimización, el objetivo es generalmente encontrar la mejor de entre varias alternativas. En algunos problemas, esto se puede hacer en un tiempo razonable visitando todas las soluciones posibles del espacio de búsqueda (ejemplo, búsqueda en una base de datos) o mediante alguna técnica que reduce el problema (ejemplo, minimización de funciones continuas). Sin embargo, existen problemas no simplificables con un espacio de búsqueda demasiado grande para explorarlo por completo con técnicas exhaustivas. En estos casos tiene sentido aplicar las *técnicas metaheurísticas*. De forma breve podemos definir las metaheurísticas como estrategias de alto nivel para explorar espacios de búsqueda. La estrategia concreta que emplean depende de la filosofía de la técnica en sí. Algunas de ellas pueden considerarse versiones "inteligentes" de algoritmos de búsqueda local (simulated annealing). Otras, sin embargo, intentan aprender la correlación entre las variables de decisión para identificar zonas de gran calidad en el espacio de búsqueda (computación evolutiva).

Los algoritmos de *Swarm intelligence* (SI) son técnicas metaheurísticas de inteligencia artificial basados en el estudio de comportamientos colectivos presentes en sistemas de la naturaleza, generalmente de carácter descentralizado y autorganizativo. Dicho comportamiento social define los movimientos de las variables de decisión en el espacio de búsqueda y las orienta hacia soluciones óptimas. La expresión de Swarm Intelligence fue introducida por Gerardo Beni, Suzanne Hackwood y Jing Wang en 1989, en el contexto de sistemas robóticos celulares.

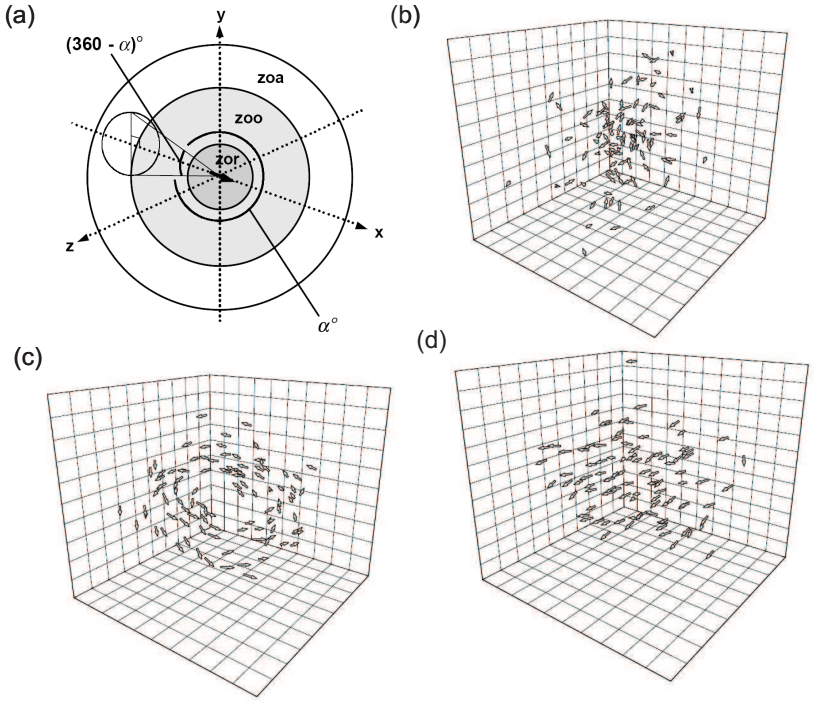


Figura 3.1: (a) Radio de influencia de un individuo, (b) colectivo Swarm, (c) colectivo Torus y (d) colectivo Dinamic/Highly Parallel Group

Los individuos deben mantener una distancia mínima entre ellos el resto del cúmulo durante todo el proceso. Por otra parte, cada individuo debe evitar una distancia excesiva con el resto del grupo, de modo que tiende a ser atraído hacia los demás individuos, evitando así ser aislado y dando lugar a diferentes vecindarios dependiendo de la fuerza de atracción. Como se puede observar en la Figura 3.1a, el radio de influencia de cada individuo se compone de una zona inicial de repulsión (zor - para evitar colisiones), una zona de orientación (zoo - el individuo dirige su movimiento) y una zona exterior de atracción (zoa - el individuo evita ser aislado). Dependiendo de la configuración de cada zona del radio individual, se generarán diferentes comportamientos y formas en los vecindarios globales.

Couzin el al. [10] identificaron una serie de comportamientos dinámicos colectivos ilustrados en la Figura 3.1:

- swarm (Figura 3.1b): agregación con cohesión pero un bajo nivel de polarización (alineamiento paralelo) entre sus miembros.
- Torus (Figura 3.1c): los individuos rotan continuamente sobre un eje vacio. La dirección de rotación es aleatoria.
- Dinamic/Highly Parallel Group (Figura 3.1d): movimiento estático en términos de intercambio de posición espacial entre los individuos.

Dentro de los algoritmos de inteligencia colectiva, se pasa brevemente a describir los de Particle Swarm Optimization y Ant Colony Optimization, que son de gran importancia por la aplicabilidad que están demostrando y la cantidad de publicaciones que vienen generando en los últimos años. Como ejemplo adicional, se reserva un pequeño espacio a los algoritmos basados en los procesos de polinización por la particularidad de su diseño.

3.1.1. Algoritmos de Colonias de Hormigas

Los algoritmos de optimización basados en colonias de hormigas o *Ant Colony Optimization* (ACO) fueron presentados por Marco Dorigo en 1992. Básicamente, se trata de una técnica probabilística para resolver problemas computacionalmente complejos que pueden ser reducidos a la búsqueda de buenos caminos en grafos. ACO se inspira directamente en el comportamiento de las colonias reales de hormigas para solucionar problemas de optimización combinatoria. Se basan en una colonia de hormigas artificiales, esto es, unos agentes computacionales simples que trabajan de manera cooperativa y se comunican mediante rastros de feromona artificiales.

Los algoritmos de ACO son esencialmente algoritmos constructivos: en cada iteración del algoritmo, cada hormiga construye una solución al problema recorriendo un grafo de construcción. Cada arista del grafo, que representa los posibles pasos que la hormiga puede dar, tiene asociada dos tipos de información que guían el movimiento de la hormiga:

- *Información heurística*, que mide la preferencia heurística de moverse desde el nodo r hasta el nodo s, o sea, de recorrer la arista a_{rs}. Se denota por η_{rs}. Las hormigas no modifican esta información durante la ejecución del algoritmo.
- *Información de los rastros de feromona artificiales*, que mide la "deseabilidad aprendida" del movimiento de r_{as}. Imita a la feromona real que depositan las hormigas naturales. Esta información se modifica durante la ejecución del algoritmo dependiendo de las soluciones encontradas por las hormigas. Se denota por τ_{rs}.

A lo largo del tiempo, el rastro de feromonas se va evaporando reduciendo así la atracción de las hormigas. Mientras un camino sea más largo, más tiempo tardarán las hormigas que tomen ese camino y más rastro de feromona se evaporará. Así, el camino más corto es el más reforzado con el olor. Sin embargo, en la filisofía del algoritmo, cuando un camino tiene mucha densidad de tráfico se suele incrementar el factor de evaporación. De este modo se reduce entrar en óptimos locales.

La información recolectada a lo largo del proceso de búsqueda se actualiza en lo que se llama la matriz de feromonas. Esta información consiste en rastros de feromonas asociados a los arcos del grafo. Ante cada toma de dicisión, una hormiga utiliza la matriz de feromonas tomando la información propia, la del nodo en el que se encuantra y la de los arcos adyacentes (feromonas). Para cada arco transitado por una hormiga, ésta deja es rastro (arco) registrado en la matriz. A continuación se muestra el pseudocódigo de un algoritmo ACO típico:

Algorithm 1 Pseudocódigo de Algoritmo ACO Canónico

```
1: procedure ACO()
2: while not stop condition do
3:     generate_new_ant
4:     update_memori; move_ant;
5:     update_pheromone_matrix_colony
6:     destroy_ant
7:     evaporate_ant_pheromones
8: end while
9: Output: Best ant found
```

Los algoritmos de colonias de hormigas han sido utilizados para obtener soluciones cercanas al óptimo en el problema del Viajante de Comercio. Tiene una ventaja principal sobre otras técnicas metaheurísticas como los Algoritmos Genéticos o el Enfriamiento Simulado y es que cuando el problema cambia dinámicamente, el algoritmo ACO puede cambiar dinámicamente tambien durante una ejecución en tiempo real. Esto es muy interesante para applicaciones como enrutado de redes o control del tráfico urbano.

Estos algoritmos se aplican con éxito a problemas de optimización combinatoria, donde la dimensión del espacio completo de soluciones es exponencial con respecto a la dimensión de la representación del problema (NP-duros). Debido a su própia naturaleza poblacional son fáciles de paralelizar. Se ha demostrado que esta metaheurística es altamente competitiva en problemas académicos complejos como: TSP, Problema de Asignación Cuadrática, Ordenación secuencial, problemas de planificación de tareas, etc.

3.1.2. Algoritmos de Cúmulos de Partículas

La técnica de los algoritmos de cúmulos de partículas o *Particle Swarm Optimization* (PSO) fue propuesta por primera vez por Kennedy y Eberhart en 1995 [27]. PSO es un algoritmo evolutivo poblacional inspirado en el comportamiento social de las bandadas de pájaros y los bancos de peces.

Un cúmulo (swarm) se compone de un conjunto de partículas (de la misma forma a una población en los EAs) representando las soluciones a un problema. En cada iteración, todas las partículas se mueven por el espacio del problema con el objetivo de encontrar una solución óptima global. Cada partícula tiene un vector que representa su posición actual y otro vector representando su velocidad (dirección de gradiente) que dirige su movimiento.

$$v_i^{k+1} = \omega \cdot v_i^k + \varphi_1 \cdot rnd_1 \cdot (pBest_i - x_i^k) + \varphi_2 \cdot rnd_2 \cdot (g_i - x_i^k) \tag{3.1}$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{3.2}$$

Las ecuaciones 3.1 y 3.2 describen la velocidad y la actualización de la posición de una partícula i en la iteración k. La Ecuación 3.1 calcula la nueva velocidad v_i para cada partícula basándose en su velocidad anterior, la posición que tenía la partícula cuando encontró el mejor valor de fitness $pBest_i$ y la posición del cúmulo en la que se encontró el mejor fitness global g_i. Cada uno de estos factores viene influenciado mediante pesos de tendencia individual o social φ_1 y φ_2 respectivamente. Finalmente, rnd_1 y rnd_2 son números aleatorios en el rango $\{0, 1\}$, y ω representa el factor inercia del movimiento individual. La Ecuación 3.2 actualiza la posición de una partícula x_i en el espacio de soluciones. En el siguiente pseudocódigo se describe el funcionamiento de un PSO canónico.

Algorithm 2 Pseudocodigo del algoritmo PSO Canónico

```
S ← SwarmInitialization()
while not stop condition do
    for each particle x_i of the swarm S do
        evaluate(x_i)
        if fitness(x_i) is better than fitness(h_i) then
            h_i ← x_i
        end if
        if fitness(h_i) is better than fitness(g_i) then
            g_i ← h_i
        end if
    end for
    for each particle x_i of the swarm S do
        v_i^{k+1} = ω · v_i^k + φ_1 · rnd_1 · (pBest_i − x_i^k) + φ_2 · rnd_2 · (g_i − x_i^k)
        x_i^{k+1} = x_i^k + v_i^{k+1}
    end for
end while
Output: best solution found
```

Los algoritmos de PSO vienen siendo aplicados con éxito a multitud de problemas de optimización combinatoria, donde la dimensión del espacio completo de soluciones es exponencial con respecto a la dimensión de la representación del problema (NP-dificil). Aunque inicialmente se formularon para trabajar con representación real para el entrenamiento de redes neuronales, están surgiendo nuevas versiones que permiten trabajar con representaciones discretas. Un ejemplo es la versión de PSO basada en espacios geometricos utilizada en este trabajo, en la cual, que se modifica el algoritmo para trabajar con soluciones binarias (véase la Sección 3.2).

3.1.3. Algoritmos Basados en la Polinización de Abejas Artificiales

Conocidos como *Artificial Bee Colony* (ABC) es uno de los algoritmos más recientes en el dominio de la inteligencia colectiva. Fue creado por Dervis Karaboga en 2005 [25], motivado por el comportamiento inteligente observado en las abejas domésticas. El autor las define tan simple como los algoritmos PSO y la Evolución Diferencial (DE) y utiliza parámetros como el tamaño de la colonia y el tamaño máximo de ciclo. ABC es un algoritmo de optimización combinatoria basado en poblaciones en el cual los individuos, llamados posiciones de comida, son modificados por las abejas artificiales. El objetivo de estas abejas es descubrir los lugares de comida con mayor nectar y finalmente el de mayor cantidad de nectar, simulando soluciones óptimas locales y el óptimo global.

En un sistema ABC, las abejas artificiales se mueven en un espacio de búsqueda multidimensional eligiendo fuentes de nectar dependiendo de su experiencia pasada y de su compañero de colmena. Algunas abejas se mueven aleatoriamente sin influencia externa (exploradoras). Cuando encuentran una fuente de nectar mayor, memorizan su posición y olvidan la anterior. De este modo, ABC combina métodos de búsqueda local y búsqueda global, intentando equilibrar el balance entre exploración y explotación del algoritmo.

Debido a su novedad, ABC aun no ha sido aplicado a problemas del ámbito industrial y la investigación se centra todavía en casos de estudio académicos.

3.2. Algoritmo de Cúmulos Basado en Espacios Geométricos

Esta novedosa versión de algoritmos de cúmulos, llamada *Geometric PSO* (GPSO), utiliza el concepto de un marco de trabajo geométrico en el que las partículas se influyen dependiendo de su distancia en el espacio. Dicha distancia será definida en función de la representación de las soluciones. Por lo tanto, para representación binaria, se utilizará la distancia de Hamming ya que en este caso, el marco geométrico pasa a ser un espacio "hammiltoniano".

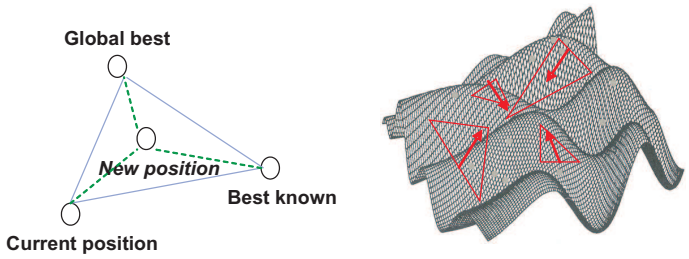


Figura 3.2: Representación de las partículas en el marco geométrico de influencia de GPSO (izquierda) y sobre el espacio de búsqueda (derecha)

Del mismo modo, para soluciones con representación entera se utilizará la distancia Manhattan y para representación continua se utilizará la distancia euclídea (véase la Figura 3.2). Una descripción más detallada de este enfoque se puede encontrar en [34]. En este trabajo utilizamos la versión con distancias de Hamming ya que los problemas abordados en los siguientes capítulos utilizan una representación binaria. Con esto conseguimos evaluar una versión no-continua de PSO obteniendo resultados prometedores.

En esta versión, la localización de cada partícula i se representa mediante un vector $x_i = \langle x_{i1}, x_{i2}, ..., x_{iN} \rangle$ tomando cada bit x_{ij} (con j en $\{1, N\}$) valores binarios 0 o 1. La característica diferenciadora de GPSO reside en el concepto de movimiento. Aquí, en vez de un factor velocidad añadido al movimiento, se utiliza un operador de cruce llamado *three-parent mask-based crossover* (3PMBCX) para generar una nueva posición de la partícula. De acuerdo con la definición de 3PMBCX dada en [34]: dados tres padres a, b y c en $\{0, 1\}^n$, se genera aleatoriamente una máscara de cruce de longitud n con los símbolos del alfabeto $\{a, b, c\}$, representando la influencia de cada padre. La partícula descendiente se genera tomando el cada elemento con el bit del padre que aparece en la máscara, bit a bit hasta generar el descendiente completo.

En Algorithm 3 se ilustra el pseudocódigo del algoritmo GPSO para espacios de Hamming. Para una partícula i, existen tres padres que toman partido en el operador (linea 13): la posición actual x_i, la mejor posición social g_i la mejor posición histórica ocupada por dicha partícula h_i. Los valores de peso $w1$, $w2$ y $w3$ indican la probabilidad de tomar valores de cada padre x_i, g_i o h_i respectivamente en la máscara del cruce. Estos valores se asocian con cada padre, representando el valor de *inercia* de la posición actual ($w1$), la influencia *social* de la mejor posición global/local ($w2$) y la influencia *individual* histórica mejor encontrada ($w3$). Una restricción del cruce geométrico fuerza a los valores $w1$, $w2$ y $w3$ a ser non-negativos y de suma 1.

Algorithm 3 Pseudocódigo del algoritmo GPSO para espacios de Hamming

```
1: S ← SwarmInitialization()
2: while not stop condition do
3:     for each particle x_i of the swarm S do
4:         evaluate(x_i)
5:         if fitness(x_i) is better than fitness(h_i) then
6:             h_i ← x_i
7:         end if
8:         if fitness(h_i) is better than fitness(g_i) then
9:             g_i ← h_i
10:        end if
11:    end for
12:    for each particle x_i of the swarm S do
13:        x_i ← 3PMBCX((x_i, w_1), (g_i, w_2), (h_i, w_3))
14:        mutate(x_i)
15:    end for
16: end while
17: Output: best solution found
```

En resumen, el algoritmo GPSO desarrollado en este estudio funciona de la siguiente forma: en una primera fase se inicializan las partículas mediante la función $SwarmInitialization()$ (Linea 1). En una segunda fase, tras la evaluación de las partículas del swarm (linea 4), se actualizan los valores históricos y sociales (lineas 5 a 10). Finalmente, se "mueven" las partículas mediante el operador 3PMBCX operator (linea 13). Además, con una cierta probabilidad, se aplica un operador de mutación (line 14) con la intención de evitar una convergencia temprana. Este proceso se repite hasta obtener una condición de parada o hasta alcanzar un cierto número de evaluaciones.

Debido a la novedad de este algoritmo, ya que fue creado en 2006, todavía no se ha aplicado a muchos problemas para evaluar su funcionamiento. Trabajos recientes han utilizado GPSO para resolver cierto tipo de Sudokus [35]. Aunque es importante matizar que este proyecto de Máster es pionero en el empleo de GPSO en la resolución de problemas reales, basándose en un trabajo publicado por el autor donde se aplica a la selección y clasificación de genes en Microarrays de ADN [15].

Capítulo 4

Selección y Clasificación de Genes en Microarrays de ADN

Tal y como se explicó en el Capítulo 1, el problema de la selección y clasificación de genes tiene una complejidad NP-dificil, por lo que se requiere el uso de técnicas metaheurísticas para su resolución. Con la intención de ofrecer una idea básica de cómo abordar este problema, en la Figura 4.1 se ilustra un esquema resumido sobre el modo en el que se extraen las características (genes) desde el Microarray inicial y cómo se evalúa el Microarray reducido resultante.

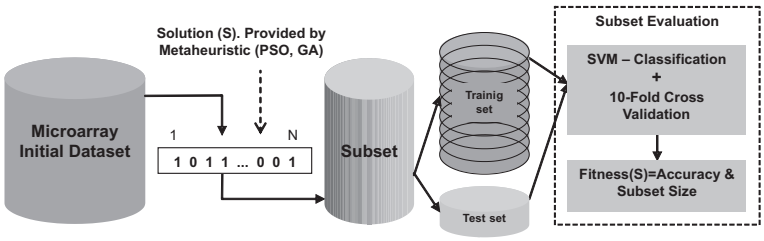


Figura 4.1: Esquema resumido sobre cómo se seleccionan los genes desde el Microarray original utilizando una partícula con codificación binaria. En una segunda fase, el subconjunto resultado es evaluado mediante el clasificador SVM y 10-fold cross validation para obtener el valor de fitness (accuracy) para dicha partícula

En una primera fase, el algoritmo metaheurístico implicado, GPSO en este caso, provee una partícula binaria (codificando una solución) en la que cada bit representa un gen. Si el bit es "1", entonces ese gen es exportado para formar parte del subconjunto, si es "0" indica que ese gen no será incluido. Por lo tanto, la longitud de la partícula será igual al número de genes en el MA inicial.

Una vez formado el subconjunto seleccionado, se le aplica el clasificador. En este trabajo se ha elegido Support Vector Machines (SVM) por ser uno de los clasificadores más sofisticados en el campo de la teoría de aprendizaje estadístico. Mediante SVM se predicen la clase a la que pertenece cierto punto asignándolo a uno de dos espacios disjuntos formados durante el entrenamiento [8]. SVM realiza principalmente clasificación binaria (2 clases): para datos linealmente separables, SVM obtiene el hiperplano que maximiza la distancia entre las muestran entrenadas y los bordes de cada clase. Para datos no separables linealmente, las muestras son extrapoladas a un espacio con una dimensión mayor de forma que esta separación es más fácil de calcular. Esta extrapolación se lleva a cabo mediante un mecanismo llamado función de *Kernel* (Figura 4.2).

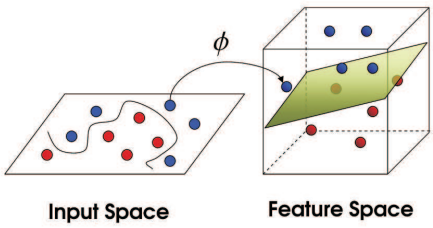


Figura 4.2: Representación del funcionamiento de la función Kernel de SVM

Una vez entrenado el subconjunto, se le aplica *10-fold cross validation* para la validación de la clasificación y la obtención de la proporción de adecuación, es decir, el error medio en la clasificación. Mediante este mecanismo, se divide el subconjunto de muestras en 10 partes iguales, utilizando 9 de estas como conjunto de entrenamiento y la parte restante como conjunto de test externo. Así, se clasifican las muestras del conjunto de test utilizando el de entrenamiento lo que arrojará un porcentaje de acierto. Esto se repite por cada uno de los 10 subconjuntos y se calcula el porcentaje de acierto total, que será reportado como valor de *accuracy*.

4.1. Función de Fitness

Puesto que la posición de una partícula x_i representa un subconjunto de genes, su evaluación tras la aplicación del clasificador y la cross validación se expresa mediante la siguiente función de fitness (Equation 4.1):

$$fitness(x) = \alpha \cdot (100/accuracy) + \beta \cdot \#features, \tag{4.1}$$

donde α y β son valores de peso, 0.75 y 0.25 respectivamente, para controlar que el valor de *accuracy* tenga prioridad sobre el número de genes ($\#features$), ya que se intentan mantener los valores altos de accuracy en el proceso de búsqueda. El objetivo consiste pues en maximizar el valor de *accuracy* y minimizar el número de genes ($\#features$). Para formalizar la función agregativa (todo minimización), el primer factor se calcula como $(100/accuracy)$.

4.2. Microarrays Utilizados

Las instancias usadas en este estudio consiste en seis bases de datos públicas referentes a experimentos de Microarray. Los valores de cada instancia fueron normalizados en el rango $\{-1, +1\}$ con el fin de mejorar el entrenamiento del clasificador y poder establecer comparativas equitativas. Dichas instancias fueron tomadas del sitio web del Kent Ridge Bio-medical Data Repository con URL `http://sdmc.lit.org.sg/GEDatasets/Datasets.html`.

- ALL-AML Leukemia, consiste en 72 experimentos con 7129 niveles de expresión de genes. Se distinguen dos clases: *Acute Myeloid Leukemia* (AML) y *Acute Lymphoblastic Leukemia* (ALL). La instancia completa contiene 25 muestras de la clase AML y 47 de ALL.

- Breast cancer, consiste en 97 experimentos con 24481 niveles de expresión. Los pacientes estudiados muestran dos clases de diagnóstico llamados *relapse* con 46 pacientes y *non-relapse* con 51.

- Colon tumor, consiste en 62 experimentos obtenidos de pacientes con 2000 niveles de expresión. Entre estos, 40 biopsias proceden de muestras de tumor (*tumors*) y 22 biopsias sanas (*normal*).

- Lung cancer, contiene 181 experimentos con 12533 niveles de expresión de genes. Se clasifica entre las clases *Malignant Pleural Mesothelioma* (MPM) y *Adenocarcinoma* (ADCA). En dichas muestras hay 31 MPM y 150 ADCA.

- Ovarian cancer, consiste en 253 experimentos con 15154 niveles de expresión. Se pretende distinguir patrones proteómicos para clasificar entre muestras cancerígenas y sanas. La instancia incluye 162 (de 253) cancerígenas y 91 sanas.

- Prostate cancer, contiene 136 experimentos con 12600 niveles de expresión. Se deben diferenciar dos clases: *tumor* con 77 $(52 + 25)$ muestras y *non-tumor* con 59 (50+9).

4.3. Algoritmos y Parámetros

El algoritmo GPSO fue programado en C++ siguiendo la arquitectura de *skeleton* de la biblioteca MALLBA [14]. Además, para realizar comparaciones en la evaluación de este algoritmo se ha implementado un algoritmo GA para resolver el mismo problema. Este algoritmo fue implementado en C++ utilizando el marco de trabajo proporcionado por la biblioteca ParadisEO [5].

El algoritmo GA desarrolla una estrategia generacional (todos los descendientes realizan reemplazo con elitismo) utilizando los siguientes operadores: selección por torneo determinista, cruce SSOCF y mutación bit-filp uniforme.

Los parámetros utilizados en ambos algoritmos son mostrados en la Tabla 4.1. Estos parámetros fueron seleccionados tras una serie de evaluaciones test sobre cada algoritmo y microarray instancia, utilizando los que mejor configuración ofrecen en términos de calidad de soluciones y esfuerzo computacional.

Tabla 4.1: Parámetros utilizados en las ejecuciones de GPSO y GA en la selección y clasificación de genes

PSO		**GA**	
Parameter	**Value**	**Parameter**	**Value**
Tamaño del cúmulo	40	Tamaño de la población	40
Número de generaciones	100	Número de generaciones	100
Tamaño del vecindario	20	Probabilidad de cruce	0.9
Probabilidad de mutación	0.1	Probabilidad de mutación	0.1
(w1, w2, w3)	(0.33, 0.33, 0.34)	-	-

Para los dos algoritmos (GPSO y GA) se utilizó el clasificador SVM ofrecido por la biblioteca LIBSVM [6] disponible en internet. Como función de Kernel se utilizó *Radial Basis Function* (RBF), utilizando una herramienta de rejilla para obtener la mejor configuración de los parámetros C y γ. Esta herramienta realiza pruebas iterativas utilizando conjuntamente dichos parámetros en cierto rango. En este trabajo consistió en los rangos $\{C = 2^{-5}, 2^{-3}, ..., 2^{15}, \gamma = 2^{-15}, 2^{-13}, ..., 2^{3}\}$. Los parámetros obtenidos para el kernel sobre cada instancia fueron:

- Colon: $C = 128{,}0$ y $\gamma = 0{,}0001220703125$
- Leukemia: $C = 8{,}0$ y $\gamma = 0{,}0001220703125$
- Breast: $C = 32{,}0$ y $\gamma = 0{,}000030517578125$
- Ovarian: $C = 128{,}0$ y $\gamma = 0{,}0001220703125$
- Lung: $C = 2{,}0$ y $\gamma = 0{,}0001220703125$
- Prostate: $C = 8{,}0$ y $\gamma = 0{,}0001220703125$

4.4. Experimentos y Resultados

Todos los experimentos fueron llevados a cabo sobre PCs con sistema operativo Linux (Suse 9.0 con kernel 2.4.19) y procesadores Pentium IV a 2.8GHz, con 512MB de RAM. Se realizaron 30 ejecuciones independientes de cada algoritmo con cada una de las 6 instancias, con esto conseguimos tener significancia estadística ya que ambos algoritmos son de naturaleza estocástica.

De estos experimentos destacamos varias observaciones, analizando el funcionamiento y robustez de los algoritmos, así como la calidad de las soluciones obtenidas, realizando además una breve justificación biológica de los genes más representativos obtenidos.

Análisis de Resultados

Desde el punto de vista del funcionamiento, ambos algoritmos obtienen resultados aceptables en pocas iteraciones, arrojando subconjuntos reducidos con altos ratios de clasificación. Sin embargo, el comportamiento es diferente. La Figura 4.3 muestra la evolución gráfica, en términos de la media del valor de fitness, de una típica ejecución de GPSO y GA. Es claramente observable que en pocas iteraciones (4 o 5) la media del fitness decrece rápidamente y termina con soluciones similares. La gran diversidad introducida en el método de inicialización provoca la aparición de rápidas y buenas soluciones junto con la convergencia temprana de los dos algoritmos. El GA obtiene generalmente medias más bajas que GPSO, cuyas soluciones mantienen una diversidad más alta.

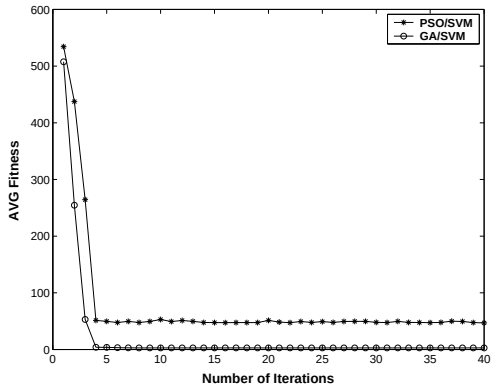


Figura 4.3: Evolución de la media del fitness (AVG_Fitness) en una ejecución típica de GPSO y GA sobre la instancia Leukemia

En la Tabla 4.2 se muestran los resultados obtenidos para cada instancia. Las columnas 2 y 3 contienen las medias de las mejores soluciones encontradas en 30 ejecuciones independientes de los algoritmos GPSO y GA respectivamente. En las columnas 4 a 8 se presentan los resultados obtenidos por cuatro métodos encontrados en la literatura, comparando así la calidad de soluciones obtenida por los algoritmos presentados. Para comparar estos resultados se utiliza el criterio estándar en la mayoría de publicaciones: el *ratio de clasificación* (*accuracy*) en términos de corrección de la clasificación (primer valor en cada celda de la tabla) y el *número de genes seleccionados* (valor entre paréntesis).

En esta comparación se puede observar que todas las soluciones obtenidas por GPSO y GA presentan un ratio de clasificación superior al 86 % en los que son comunes subconjuntos de 4 genes e incluso menores. En este trabajo se mejoran todos los resultados existentes (según nuestro conocimiento) excepto en un caso reportado en [20] en el que presenta subconjuntos de 2 genes. Posiblemente, el efecto de los operadores de cruce especializados aporten calidad a las soluciones en la evolución, ya que tanto el cruce 3PMBCX utilizado en GPSO como el cruce SSOCF utilizado en GA conservan los patrones de genes signi-

Tabla 4.2: Comparación de los algoritmos evaluados con trabajos relacionados del estado del arte. Los resultados con mejores accuracy son remarcados. Las celdas sin valor conocido se ocupan con el carácter '-'

Dataset	$GPSO_{SVM}$	GA_{SVM}	[24]	[20]	[21]	[31]	[52]
Luekemia	97.38(3)	97.27(4)	-	100(2)	100(25)	-	87.55(4)
Breast	86.35(4)	95.86(4)	-	-	-	-	79.38(67)
Colon	100(2)	100(3)	94.12(37)	98.0(4)	99.41(10)	85.48(-)	93.55(4)
Lung	99.00(4)	99.49(4)	-	-	-	-	98.34(6)
Ovarian	99.44(4)	98.83(4)	-	-	-	99.21(75)	-
Prostate	98.66(4)	98.65(4)	88.88(20)	-	-	-	-

ficativos durante el proceso. En segundo lugar, la parametrización sistemática del Kernel RBF mejora la fase entrenamiento de los datos y por consiguiente la validación mediante *10 k fold cross-validation*.

Si se centra en la comparación en los algoritmos GPSO y GA, se obtienen resultados similares. En general, el GA obtiene un mayor número de mejores soluciones (Hits), aunque la mejor clasificación fue obtenida por GPSO para la instancia de Colon (100 % accuracy y 2 genes, Tabla 4.2). Desde el punto de vista de la accuracy media en todas las ejecuciones independiente, GPSO obtiene un mejor funcionamiento, si bien la diferencia respecto a GA es no es significante (como muestra la Figura 4.3).

Análisis de la Robustez

Uno de los criterios más importantes en la evaluación de cualquier nuevo algoritmo propuesto, es la calidad del algoritmo y su capacidad en generar soluciones similares cuando se ejecuta varias veces. Este factor es incluso crítico en el caso de las metaheurísticas, siendo el caso de este trabajo. Para examinar la robustez de los algoritmos evaluados, en todas las instancias y en todas las ejecuciones podemos encontrar soluciones similares (no idénticas). Sin embargo, se debe mencionar que el total de accuracy y el número de genes en todos los casos no se desvían (por desviación típica) los unos de otros en más de 5.5. La Tabla 4.3, muestra los resultado al ejecutar GA y GPSO en términos estadísticos, destacando la mejor solución encontrada (*Best*), la media (*Mean*) y la desviación típica (*Std. Dev.*) en las treinta ejecuciones independientes.

Tabla 4.3: Comparaciones de resultados en términos estadísticos de $GPSO_{SVM}$ y GA_{SVM}

Dataset	$GPSO_{SVM}$			GA_{SVM}		
	Best	Mean	Std Dev.	Best	Mean	Std Dev.
Leukemia	100(3)	97.38(3)	3.80	100(4)	97.27(4)	3.82
Breast	90.72(4)	86.35(4)	4.11	100(4)	95.86(4)	5.33
Colon	100(2)	100(2)	0.0000	100(3)	100(3)	0.0000
Lung	99.44(4)	99.00(4)	0.50	100(4)	99.49(4)	0.41
Ovarian	100(4)	99.44(4)	0.38	100(4)	98.83(4)	3.18
Prostate	100(4)	98.66(4)	1.14	100(4)	98.65(4)	3.24

Breve Análisis Biológico de los Genes Encontrados

Finalmente, en la Tabla 4.4 se muestra un breve resumen de los mejores subconjuntos de genes encontrados para cada instancia. Todos los subconjuntos obtienen accuracies cercanas al 100 % con el mínimo número de genes. Se hace hincapié en que aparentemente (bajo nuestro conocimiento) algunos de los genes seleccionados no lo habían sido en ningún estudio de la literatura. En este sentido, se ofrece a continuación una breve descripción biológica de algunos de los genes obtenidos con más frecuencia, los cuales son utilizados actualmente en el diseño de medicamentos y en el tratamiento de cánceres.

Tabla 4.4: Subconjuntos de genes con 100 % accuracy obtenidos con mayor frecuencia por GPSO y GA

Dataset	$GPSO_{SVM}$		GA_{SVM}	
Leukemia	100(3)	*U39226_at, L12052_at, X99101_at*	100(4)	*Z26634_at, HG870-HT870_at X52005_at, L02840_at*
Breast	90.72(4)	*NM_012269, NM_002850 AL162032, AB022847*	100(4)	*NM_005014, AF060168 NM_021176, NM_013242*
Colon	100(2)	*U29092, M55543*	100(3)	*M90684, M94132 X62025*
Lung	99.44(4)	*31820_at, 33389_at 39057_at, 40772_at*	100(4)	*31573_at, 33226_at 36245_at, 37076_at*
Ovarian	100(4)	*MZ49.784115, MZ3546.2884 MZ4362.0866, MZ9159.3641*	100(4)	*MZ420.40671, MZ825.16557 MZ1024.6857, MZ1166.0749*
Prostate	100(4)	*35106_at, 35869_at 36754_at, 37107_at*	100(4)	*41447_at, 34299_at 39556_at, 39813_s_at*

- Gene *L12052_at* se trata de "CAMP phosphodiesterase mRNA, 3' end" utilizado en medicamentos como Anagrelide o Milrinone. Concretamente la Anagrelide se utiliza en el tratamiento de la trombosis y se demostró ser efectiva en el tratamiento de pacientes con cierto tipo de leucemia como chronic myeloid leukemia [40]. Este gen pertenece a un conjunto de 3 genes (obtenidos de la instancia Leukemia en la Tabla 4.4) con 100 % accuracy, seleccionada por $GPSO_{SVM}$.

- Gene *AB022847* consiste en "Solute carrier family 6 (neurotransmitter transporter, noradrenalin), member 2" localizado en la membrana del plasma. Drogas actuales como Radaxafine, Amphetamine o Venlafaxine están asociadas con este gen. Concretamente, la Venlafaxina fue utilizado por primera vez por Wyeth en 1993 como antidepresivo. Además, se utiliza en el tratamiento de flashes en superviviente de cancer de pecho [32]. Este gen pertenece a un conjunto de 3 genes (obtenidos de la instancia Breast en la Tabla 4.4) con 95.8763 % accuracy, seleccionada por $GPSO_{SVM}$.

- Gene *36245_at* se trata de "5-hydroxytryptamine (serotonin) receptor 2B" localizado en la membrana de plasma humano. Se utiliza en la fabricación de varias drogas como Risperidona, Blonanserin y Mirtazapina. Algunos estudios consideran la Mirtazapina como la primera elección para el tratamiento de la ansiedad y la depresión después del transplante de pulmón. Este gen pertenece a un conjunto de 4 genes (obtenidos de la instancia Lung en la Tabla 4.4) con 100 % accuracy, seleccionada por GA_{SVM}.

Capítulo 5

Gestión de la Localización de Terminales Móviles

En este capítulo, se describen los pasos seguidos en la aplicación del algoritmo GPSO al problema de la "gestión de la localización de terminales móviles en redes GSM" (LA), introducido en el Capítulo 2. Sin más preámbulos, pasamos a la descripción del problema y la función de evaluación utilizada.

Para ilustrar el funcionamiento en LA, se asume la configuración de celdas de la red mostrada en la Figura 5.1, donde las celdas de gestión son marcadas en color gris. En esta ilustración, las celdas marcadas con 'N' pertenecen a los vecindarios de por lo menos tres celdas de gestión, como se muestra en las figuras 5.1a-5.1c. El número de vecinos para la celda 'X' es 25, 17, y 22 para las figuras 5.1a-5.1c respectivamente.

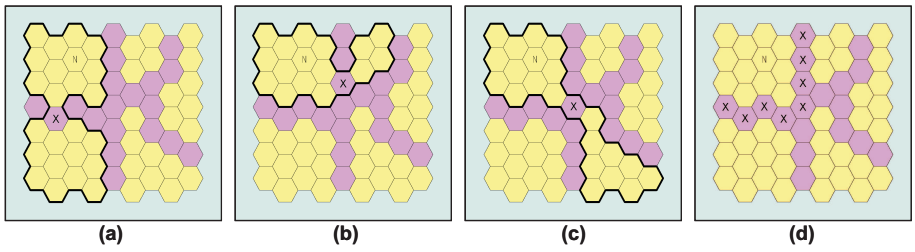


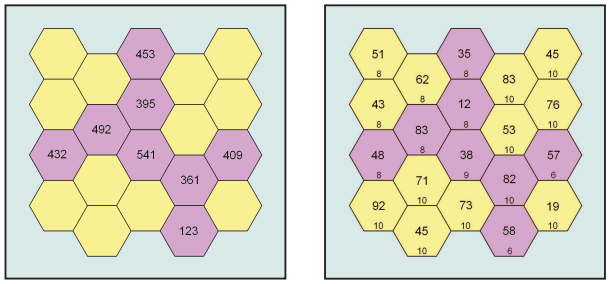
Figura 5.1: Instancia de celdas de no gestión (non-paging) pertenecientes a más de un vecindario

De esta forma, la celda 'N' es parte de por lo menos tres vecindarios con 25, 17, y 22 celdas. Considerando el peor caso, el factor de vecindario de la celda 'N' es 25 (máximo valor de 25, 17 y 22). Sin embargo, si una celda de no gestión forma parte de más de dos vecindarios, como en la Figura 5.1, se debe realizar el cálculo para todos, siendo considerado el valor máximo de todos los factores de vecindario para dicha celda.

Por ejemplo, la celda marcada con 'N' forma parte de los vecindarios de todas las celdas de gestión marcadas con 'X' en la Figura 5.1d. Ahora, para calcular el coste total de la red de gestión, se modifica la función de evaluación de manera que:

$$Cost = \beta \times \sum_{i \epsilon S} N_{LU}(i) + \sum_{i=0}^{N} N_P(i) \times V(i) \tag{5.1}$$

donde, $N_{LU}(i)$ es el número de actualizaciones de posición para la celda i, $N_P(i)$ es el número de llamadas recibidas para dicha celda i, $V(i)$ es su factor de vecindad, S es el conjunto de celdas definido como celdas de gestión y N es el número total de celdas de la red.



(a) (b)

Figura 5.2: Ejemplo con el número de (a) actualizaciones de posición y (b) llamadas recibidas por la red

Para clarificar esta explicación, podemos considerar como ejemplo la configuración de red mostrada en la Figura 5.2. En este caso, el número de actualizaciones para cada celda de gestión en la Figura 5.2a aparece al centro de cada celda. Así, el número total de actualizaciones para esta configuración será:

Total number of location update $= 453 + 395 + 541 + 492 + 432 + 361 + 409 + 123 = 3206$

En el calculo del coste de búsqueda, inicialmente se debe calcular el factor de vecindad para cada celda, ya sea celda de gestión o no. Tras esto se calcula el coste total de búsqueda. En la Figura 5.2b aparecen dos números por celda: El número en el centro determina la cantidad de llamadas entrantes, mientras que el número que aparece cerca de los bordes indica el factor de vecindad de cada celda. De este modo, el número de transacciones para esta red se obtiene:

Total number of paging transaction $= 51 \times 8 + 43 \times 8 + 48 \times 8 + 92 \times 10 + 62 \times 8 + 83 \times 8 + 71 \times 10 + 45 \times 10 + 35 \times 8 + 12 \times 8 + 38 \times 9 + 73 \times 10 + 83 \times 10 + 53 \times 10 + 82 \times 10 + 58 \times 6 + 45 \times 10 + 76 \times 10 + 57 \times 6 + 19 \times 10 = 10076$

Si cada transacción cuesta 1 unidad y cada actualización cuesta 10 unidades, es decir, $\beta = 10$, el coste total de la red será:

$$Total\ cost = 3206 \times 10 + 10076 = 42136$$

5.1. Experimentos y Resultados

En estos experimentos, se evalúa el algoritmo GPSO en la resolución del problema de LA y se realizan comparaciones respecto al un algoritmo basado en redes neuronales de Hopfield hibridado con una técnica de Ball Dropping (HNN+BD).

Los algoritmos se ejecutaron utilizando doce instancias con diferentes configuraciones de redes celulares cuadradas. Dependiendo del número de celdas de cada red, se clasifican en grupos de dimensiones: 4×4, 6×6, 8×8 y 10×10. Todas estas instancias fueron generadas exclusivamente para este trabajo y están disponibles en la dirección URL `http://neo.lcc.uma.es/la`.

Para cada algoritmo se realizaron diez ejecuciones independientes con cada instancia de red. Las observaciones realizadas en base a estos experimentos se comentan en las siguientes secciones.

5.1.1. Efecto Segmentación de la Red

Como era de esperar, las configuraciones óptimas para todas las redes muestran una configuración particionada en pequeñas subredes. Como ejemplo, se pueden observar claramente los resultados obtenidos para las instancias de redes 8×8 en la Figura 5.3.

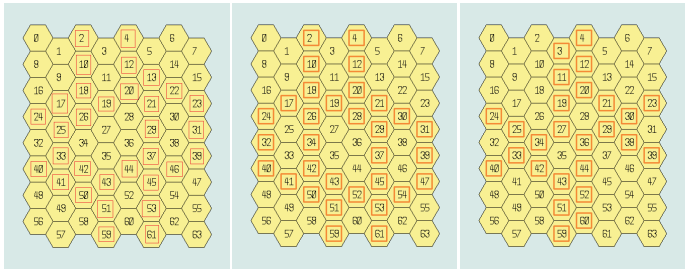


Figura 5.3: Soluciones para las redes 8×8. En la leyenda se muestra el mejor coste encontrado y el algoritmo que obtiene dicha solución para cada red

5.1.2. Análisis de la Robustez

Del mismo modo al problema estudiado en el capítulo anterior, se realiza un análisis de la robustez de los algoritmos.

Para la gran mayoría de las instancias y en todas las ejecuciones, los algoritmos encuentran las mismas soluciones, aunque en diferentes iteraciones. Sólo en ciertos casos, se encuentran soluciones similares (no idénticas). Sin embargo, se puede mencionar que los costes totales de cada red en todos los casos no se desvían en más del 0.42 %.

La Tabla 5.1 muestra los resultados obtenidos al ejecutar los algoritmos sobre la primera instancia de dimensión 4×4 (Test-Network-1). Se puede observar que ambos algoritmos encuentran las mismas soluciones pero en diferentes número de iteraciones. Esto es lógico pues se tratan de algoritmos de base heurística. Por otra parte, para instancias de redes más complejas, los algoritmos GPSO y HNN-BD obtienen resultados diferentes. En la Tabla 5.2 se presentan los resultados respecto a una red 8×8 (instancia Test-Network-9). El mejor resultado de las 10 ejecuciones se muestra sombreado y se calcula el porcentaje de desviación con respecto a éste en cada fila. Por ejemplo, el coste obtenido por GPSO en la primera ejecución es un 0.37 % más elevado que el mejor obtenido.

Tabla 5.1: Robustez sobre la instancia Test-Network-1

HNN+BD				GPSO			
No.	No. It	Energy	Dev. FBE	No.	No. It	Fitness	Dev. FBE
1	20	98535	% 0.00	1	7	98535	% 0.00
2	27	98727	% 0.19	2	4	98535	% 0.00
3	23	98535	% 0.00	3	5	98535	% 0.00
4	41	98535	% 0.00	4	10	98535	% 0.00
5	44	98727	% 0.19	5	7	98535	% 0.00
6	32	98535	% 0.00	6	5	98535	% 0.00
7	41	98535	% 0.00	7	6	98535	% 0.00
8	55	98535	% 0.00	8	5	98535	% 0.00
9	32	99072	% 0.54	9	8	98535	% 0.00
10	41	98535	% 0.00	10	7	98535	% 0.00

Tabla 5.2: Robustez sobre la instancia Test-Network-9

HNN+BD				GPSO			
No.	No. It	Energy	Dev. FBE	No.	No. It	Fitness	Dev. FBF
1	82	264824	% 0.23	1	105	265192	% 0.37
2	135	265324	% 0.42	2	131	264353	% 0.05
3	123	264204	% 0.00	3	133	264316	% 0.00
4	170	264353	% 0.05	4	141	264316	% 0.04
5	122	264786	% 0.22	5	180	264353	% 0.05
6	153	264204	% 0.00	6	61	264204	% 0.00
7	84	264786	% 0.22	7	263	264316	% 0.04
8	84	264965	% 0.28	8	201	264316	% 0.04
9	185	264353	% 0.05	9	250	264204	% 0.00
10	118	265104	% 0.34	10	60	264786	% 0.22

5.1.3. Comparación Entre GPSO y HNN+BD

En la Tabla 5.3 se pueden observar los resultados obtenidos por los dos algoritmos estudiados. La primera columna contiene el número y dimensión (entre paréntesis) de cada instancia. Para cada algoritmo evaluado se presentan tres valores: el mejor coste (Best, de las 10 ejecuciones), el coste medio (Aver.) y el porcentaje de desviación con respecto al mejor coste (Dev.).

Como se desprende de los resultados, los dos algoritmos arrojan soluciones similares para casi todas las instancias, aunque se presentan algunas diferencias para las instancias mayores. Por ejemplo, GPSO consigue mejores resultados para las instancias Test-Network 7 y 10, mientras que HNN+BD consigue ganar en la instancia Test-Network 11. Además, se puede comprobar que el porcentaje de desviación con respecto al mejor es generalmente más bajo para GPSO, especialmente para las redes más pequeñas. Este comportamiento nos lleva a pensar que GPSO es más robusto que HNN+BD.

Tabla 5.3: Resultados en coste total obtenidos por GPSO y HNN+BD

Test Network	HNN+BD			GPSO		
No. (Dim.)	Best	Aver.	Dev.	Best	Aver.	Dev.
1 (4×4)	98,535	98,627	0.09 %	98,535	98,535	0.00 %
2 (4×4)	97,156	97,655	0.51 %	97,156	97,156	0.00 %
3 (4×4)	95,038	95,751	0.75 %	95,038	95,038	0.00 %
4 (6×6)	173,701	174,690	0.56 %	173,701	174,090	0.22 %
5 (6×6)	182,331	182,430	0.05 %	182,331	182,331	0.00 %
6 (6×6)	174,519	176,050	0.87 %	174,519	175,080	0.32 %
7 (8×8)	308,929	311,351	0.78 %	308,401	310,062	0.53 %
8 (8×8)	287,149	287,149	0.00 %	287,149	287,805	0.22 %
9 (8×8)	264,204	264,695	0.18 %	264,204	264,475	0.10 %
10 (10×10)	386,351	387,820	0.38 %	385,972	387,825	0.48 %
11 (10×10)	358,167	359,036	0.24 %	359,191	359,928	0.20 %
12 (10×10)	370,868	374,205	0.89 %	370,868	373,722	0.76 %

Otra diferencia obvia entre HNN+BD y GPSO reside en el funcionamiento interno de cada algoritmo. Esto es fácilmente observable en la Figura 5.4, donde se presentan gráficamente las trazas de los mejores valores de cada iteración en la evolución de los algoritmos GPSO y HNN+BD. Cada gráfica corresponde a una de las doce instancias. El algoritmo GPSO muestra el típico comportamiento en una metaheurística evolutiva, es decir, el algoritmo tiende a converger desde las soluciones de la población inicial hasta alcanzar valores de fitness óptimos. Gráficamente, el funcionamiento de GPSO se representa mediante una curva monótona decreciente (minimización). Por otra parte, HNN+BD realiza una estrategia de búsqueda diferente, ya que desde la inicialización, se producen frecuentes reinicializaciones (sacudidas) con el propósito de intensificar y diversificar gradualmente la búsqueda. Estas "sacudidas" son producidas por la fase de Ball Dropping cuando el algoritmo detecta que no se producen mejoras en cierto intervalo de iteraciones. Evidentemente, tal y como muestra la Figura 5.4, el número de sacudidas es mayor en instancias mayores, puesto que el número de iteraciones necesarios para la convergencia es también mayor. Gráficamente, este comportamiento produce intervalos intermitentes de picos y valles a lo largo de la línea de evolución.

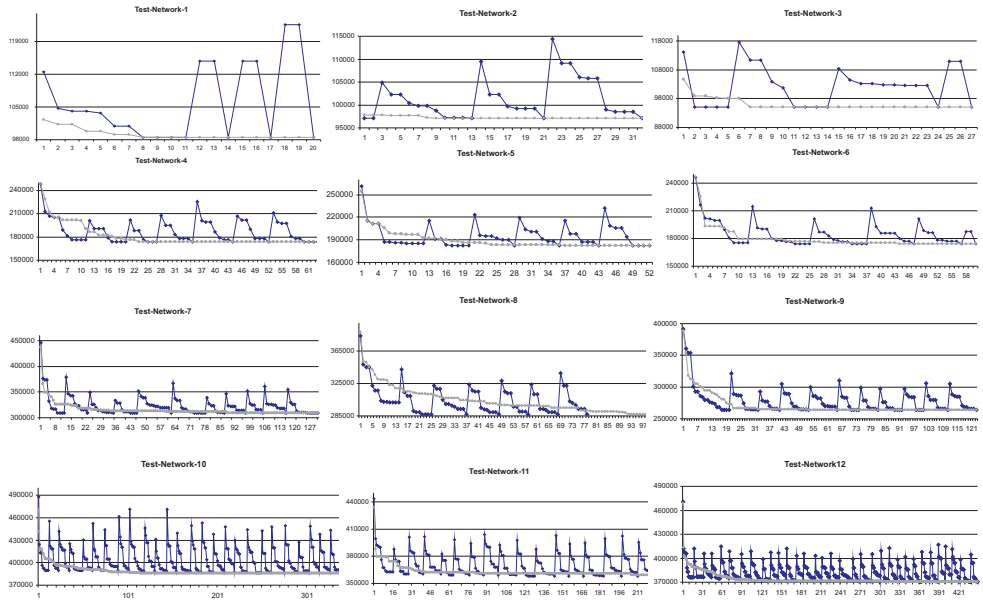


Figura 5.4: Trazas de evolución en cada instancia. Cada gráfica registra el nivel de energía obtenido en la evolución de HNN+BD (línea negra con picos y valles) y el valor de fitness en la evolución de GPSO (curva cóncava gris)

Finalmente, el tiempo requerido para llevar a cavo una ejecución con HNN+BD o GPSO supone pocos segundos e incluso el número de iteraciones para la convergencia en ambos algoritmos es similar. Sin embargo, se debe diferenciar los aspectos relacionados con el diseño. En HNN+BD, se utiliza información sobre el problema (LA) en el diseño y la construcción. La función de coste total es directamente codificada en el diseño de la red neuronal, unido a las tareas específicas que realiza la técnica de ball dropping dependiendo de la topología de la red. Esta característica hace de HNN+BD un algoritmo completamente dependiente del problema. Por el contrario, GPSO necesita un conocimiento mínimo del problema, únicamente la representación de soluciones y la función de fitness, para resolver el problema de manera igualmente eficiente. Por lo tanto, el diseño de GPSO tiene menos dependencia respecto del problema y puede ser utilizado (con mínimos cambios) en la resolución de problemas similares.

Capítulo 6

Conclusiones

En el presente trabajo se han llevado a cabo varias tareas. En primer lugar, se ha ofrecido una introducción de los problemas abordados, consistiendo en la "selección y clasificación de genes en Microarrays de ADN", del campo de la Bioinformática y la "gestión de localización de terminales móviles en redes GSM", del campo de las Telecomunicaciones. Se han elegido estos problemas por su relevancia en la investigación actual y el interés que suscita por parte de la comunidad científica. Como eje central del trabajo se propone la utilización del algoritmo Geometric Particle Swarm Optimization, una técnica metahurística bioinspirada basada en los principios de la inteligencia colectiva. Un resultado directo del estudio en este tipo de algoritmos desembocó en la publicación de un artículo en número especial de revista [14].

- Enrique Alba, Gabriel Luque, Jose Garcia-Nieto, Guillermo Ordonez, Guillermo Leguizamon, MALLBA: a software library to design efficient optimisation algorithms, *Int. J. of Innovative Computing and Applications (IJICA)*, vol 1, N. 1, pp. 74-85, 2007.

En este sentido, se ha realizado un repaso muy general por las técnicas metaheurísticas basadas en inteligencia colectiva, haciendo hincapié sobre los algoritmos más conocidos (ACO y PSO), además de presentar técnicas de última generación como ABC. Tras esta visión global, se profundiza en el diseño del algoritmo GPSO, el cual será utilizado en los experimentos realizados posteriormente.

Continuando con la resolución de los problemas introducidos en la primera parte, en el Capítulo 4 se describen los pasos seguidos en la aplicación de GPSO al problema de la selección y clasificación de genes en Microarrays de ADN. Se define la función de evaluación utilizada, las instancias de Microarray, la parametrización de los algoritmos y los experimentos realizados. En este caso se puede concluir que el nuevo algoritmo GPSO resuelve satisfactoriamente el problema, consiguiendo buenos resultados en comparación con otras técnicas (GAs) y descubriendo subconjuntos de genes cuya significancia se justifica mediante análisis biológicos.

Este trabajo se fundamenta en los resultados publicados en dos artículos de congresos internacionales en la sección de bioinformática [15] [17].

- Enrique Alba, José García-Nieto, Laetitia Jourdan and El-Ghazali Talbi, Gene Selection in Cancer Classification using PSO/SVM and GA/SVM Hybrid Algorithms. In proceedings of the *IEEE Congress on Evolutionary Computation CEC-07*, Singapore, Sep, 2007.

- Enrique Alba, José García-Nieto, Laetitia Jourdan and El-Ghazali Talbi, A comparison of PSO and GA approaches for gene selection and classification of microarray data. In proceedings of the *9th annual conference on Genetic and evolutionary computation* (GECCO'07), ACM Press, pp. 427–427, London, UK, 2007.

En cuanto al problema de la gestión de la localización de terminales móviles en redes GSM, se puede afirmar que el algoritmo GPSO constituye una técnica adecuada para la resolución de este tipo de problemas combinatorios, ya que con simples modificaciones en la representación de las soluciones y la función de evaluación, consigue resultados cercanos al óptimo, e incluso llega a batir a los algoritmos diseñados expresamente para este problema. Actualmente, el alumno está trabajando en la difusión de estos experimentos mediante la escritura de un artículo que será enviado a un congreso internacional dedicado a las técnicas metaheurísticas en la resolución de problemas combinatorios.

Por lo tanto, en este trabajo se propone el uso de los algoritmos basados en inteligencia colectiva, en concreto el algoritmo Geometric Particle Swarm Optimization, para la resolución de problemas de naturaleza combinatoria compleja. Esta propuesta se avala con una serie de experimentos cuantitativos y comparaciones con otras técnicas, resultando factibles y competitivos.

Bibliografía

[1] D.P. Agrawal and Q-A. Zeng. *Introduction to Wireless and Mobile Systems*. Thomson Brooks/Cole Inc, 2003.

[2] Alon, N. Barkai, D.Ñotterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci*, 96:6745–6750, 1999.

[3] M. Banerjee, S. Mitra, and H. Banka. Evolutionary rough feature selection in gene expression data. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, 37(4):622–632, July 2007.

[4] P. Brown and D. Botstein. Exploring the New World of the Genome with DNA Microarrays. *Nature Genetics*, 21:33–37, 1999.

[5] S. Cahon, E-G. Talbi, and N. Melab. Paradiseo: A framework for parallel and distributed metaheuristics. In *Proc. of Internationa the Parallel and Distributed Processing Symposium*, pages 144–155, 2003.

[6] C-C. Chang and C-J. Lin. LIBSVM: A library for support vector machines. http://www.csie.ntu.edu.tw/ cjlin/libsvm, 2002.

[7] C.H. Chu, G. Premkumar, and H. Chou. Digital data networks design using genetic algorithms. *European Journal of Operational Research*, 127(1):140–158, 2000.

[8] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[9] C. Cotta, A. Mendes, V. Garcia, P. França, and P. Moscato. Applying Memetic Algorithms to the Analysis of Microarray Data. In *Proceedings of the Applications of Evolutionary Computing*, volume 2611 of *Lecture Notes in Computer Science*, pages 22–32, Berlin, 2003. Springer-Verlag.

[10] I.D. Couzin, J. Krause, R. James, G.D. Ruxton, and N.R Franks. Collective memory and spatial sorting in animal groups. *Journal of Theoretical Biology*, 218:1–11, 2002.

[11] K. Deb and A. R. Reddy. Classification of two-class cancer data reliably using evolutionary algorithms. Technical report, KanGAL Report No. 2003001, 2003.

[12] I. Demirkol, C. Ersoy, M.U. Caglayan, and H. Delic. Location area planning in cellular networks using simulated annealing. In *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM 2001*, volume 1, pages 13–20, 2001.

[13] Marco Dorigo and Gianni Di Caro. The ant colony optimization metaheuristic. pages 11–32, 1999.

[14] Jose Garcia-Nieto Guillermo Ordonez Guillermo Leguizamon Enrique Alba, Gabriel Luque. Mallba: a software library to design efficient optimisation algorithms. *Int. J. of Innovative Computing and Applications 2007 (IJICA)*, 1(1):74–85, 2007.

[15] Laetitia Jourdan Enrique Alba, José Garcá-Nieto and El-Ghazali Talbi. Gene Selection in Cancer Classification using PSO/SVM and GA/SVM Hybrid Algorithms. In *IEEE Congress on Evolutionary Computation CEC-07*, Singapore, Sep 2007.

[16] Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. Technical report, 4, US Air Force School of Aviation Medicine, Randolph Field, TX, 1951.

[17] José García-Nieto, Enrique Alba, Laetitia Jourdan, and El-Ghazali Talbi. A comparison of PSO and GA approaches for gene selection and classification of microarray data. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 427–427, New York, NY, USA, 2007. ACM Press.

[18] R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[19] Gavin J. Gordon, Roderick V. Jensen, Li-Li Hsiao, Steven R. Gullans, Joshua E. Blumenstock, Sridhar Ramaswamy, William G. Richards, David J. Sugarbaker, and Raphael Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Res*, 62:4963–4967, 2002.

[20] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[21] E. B. Huerta, Bèatrice Duval, and Jin-Kao Hao. A Hybrid GASVM Approach for Gene Selection and Classification of Microarray Data. In Franz

Rothlauf, Jürgen Branke, Stefano Cagnoni, Ernesto Costa, Carlos Cotta, Rolf Drechsler, Evelyne Lutton, Penousal Machado, Jason H. Moore, Juan Romero, George D. Smith, Giovanni Squillero, and Hideyuki Takagi, editors, *Lecture Notes in Computer Science of EvoWorkshops*, volume 3907, pages 34–44. Springer, 2006.

[22] L. Jourdan, C. Dhaenens, and E-G Talbi. A genetic algorithm for feature selection in data-mining for genetics. In *Proceedings of the 4th Metaheuristics International ConferencePorto (MIC'2001)*, pages 29–34, Porto, Portugal, 2001.

[23] L. Jourdan, C. Dhaenens, and E-G. Talbi. Rules extraction in linkage disequilibrium mapping with an adaptive genetic algorithm. In *European Conference on Computational Biology (ECCB) 2003*, pages 29–32, 2003. Paris, France.

[24] Juliusdottir, D. Corne, E. Keedwell, and A.Ñarayanan. Two-phase EA/K-NN for feature selection and classification in cancer microarray datasets. In *CIBCB*, pages 1–8, 2005.

[25] D. Karaboga. An idea based on honey bee swarm for numerical optimization. Tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[26] J. Kennedy and R. Eberhart. Particle Swarm Optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, Perth, Australia, Nov 1995.

[27] J. Kennedy and R. Eberhart. Particle Swarm Optimization. In *Proc. of the IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.

[28] J. Kohavi and G. H. John. The wrapper approach. In *Feature Selection for Knowledge Discovery and Data Mining*, pages 33–50, 1998.

[29] Li L, Weinberg CR, Darden TA, and Pedersen LG. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the ga/knn method. *Bioinformatics*, 17(12):1131–1142, Dec 2001.

[30] Y.-B. Lin and I. Chlamatac. *Wireless and Mobile Network Architecture*. John Wiley and Sons, 2001.

[31] B. Liu, Q. Cui, T. Jiang, and S. Ma. A combinational feature selection and ensemble neural network method for classification of gene expression data. *BMC Bioinformatics*, 5:136–148, 2004.

[32] C. L. Loprinzi. Venlafaxine in management of hot flashes in survivors of breast cancer: a randomised controlled trial. *The Lancet*, 356(9247):2059–2063, 2000.

[33] F. Luna, E. Alba, A. J. Nebro, and S. Pedraza. Evolutionary algorithms for real-world instances of the automatic frequency planning problem in GSM networks. In *Seventh European Conference on Evolutionary Computation in Combinatorial Optimization (EVOCOP 2007)*, volume 4446 of *LNCS*, pages 108 – 120, 2007.

[34] A. Moraglio, C. Di Chio, and R. Poli. Geometric Particle Swarm Optimization. In *10th European conference on Genetic Programming (EuroGP 2007)*, volume 4445 of *Lecture Notes in Computer Science*. Springer, Abril 2007.

[35] A. Moraglio and J. Togelius. Geometric particle swarm optimization for the sudoku puzzle. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007)*, London, UK, July 2007. ACM Press.

[36] M.Ñarendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Trans. on Computer*, 26:917–922, 1977.

[37] A.C. Pease, D. Solas, E. Sullivan, M. Cronin, C. P. Holmes, and S. Fodor. Light-generated oligonucleotide arrays for rapid dna sequence analysis. In *Proc. Natl. Acad. Sci.*, volume 96, pages 5022–5026, USA, 1994.

[38] E. F. Petricoin, Ali M. Ardekani, Ben A. Hitt, Peter J. Levine, Vincent A. Fusaro, Seth M. Steinberg, Gordon B. Mills, Charles Simone, David A. Fishman, Elise C. Kohn, and Lance A. Liotta. Use of proteomic patterns in serum to identify ovarian cancer. *The Lancet*, 359:572–577, 2002.

[39] Qi Shen, Wei Min Shi, Wei Kong, and Bao Xian Ye. A combination of modified particle swarm optimization algorithm and support vector machine for gene selection and classification. *Talanta*, 2006.

[40] R. T. Silver. Anagrelide is effective in treating patients with hydroxyurea-resistant thrombocytosis in patients with chronic myeloid leukemia. *Leukemia*, 19(3):39–43, 2005.

[41] D. Singh, P. Febbo, K. Ross, D. Jackson, J. Manola, C. Ladd, P. Tamayo, A. Renshaw, A. D'Amico, and J. Richie. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1:203–209, 2002.

[42] Riky Subrata and Albert Y. Zomaya. A comparison of three artificial life techniques for reporting cell planning in mobile computing. *IEEE Trans. Parallel Distrib. Syst.*, 14(2):142–153, 2003.

[43] Riky Subrata and Albert Y. Zomaya. Evolving cellular automata for location management in mobile computing networks. *IEEE Trans. Parallel Distrib. Syst.*, 14(1):13–26, 2003.

[44] J. Taheri and A.Y. Zomaya. The use of a hopfield neural network in solving the mobility management problem. In *IEEE/ACS International Conference on Pervasive Services, ICPS 2004*, pages 141–150, Jul 2004.

[45] J. Taheri and A.Y. Zomaya. A genetic algorithm for finding optimal location area configurations for mobility management. In *30th Anniversary of the IEEE Conference on Local Computer Networks (LCN)*, pages 568–577, Nov 2005.

[46] Javid Taheri and Albert Y. Zomaya. A genetic algorithm for finding optimal location area configurations for mobility management. In *LCN '05: Proceedings of the The IEEE Conference on Local Computer Networks 30th Anniversary*, pages 568–577, Washington, DC, USA, 2005. IEEE Computer Society.

[47] Javid Taheri and Albert Y. Zomaya. A simulated annealing approach for mobile location management. In *IPDPS '05: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Workshop 6*, page 194, Washington, DC, USA, 2005. IEEE Computer Society.

[48] Javid Taheri and Albert Y. Zomaya. Clustering techniques for dynamic mobility management. In *MobiWac '06: Proceedings of the international workshop on Mobility management and wireless access*, pages 10–17, New York, NY, USA, 2006. ACM Press.

[49] Laura J. van 't Veer, Hongyue Dai, Marc J. van de Vijver, Yudong D. He, Augustinus A. M. Hart, Mao Mao, Hans L. Peterse, Karin van der Kooy, Matthew J. Marton, Anke T. Witteveen, George J. Schreiber, Ron M. Kerkhoven, Chris Roberts, Peter S. Linsley, René Bernards, and Stephen H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, 2002.

[50] Hsiao-Kuang Wu, Ming-Hui Jin, Jorng-Tzong Horng, and Chen-Yi Ke. Personal paging area design based on mobile's moving behaviors. In *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2001*, volume 1, pages 21–30, 2001.

[51] J. Yang and V. Honavar. *Feature Extraction, Construction and Selection: A Data Mining Perspective*, chapter Feature Subset Selection Using a Genetic Algorithm, pages 177–136. 1998.

[52] L. Yu and H. Liu. Redundancy based feature selection for microarry data. In *Proceedings of International Conference on Knowledge Discovery and Data Mining*, pages 22–25, Seattle, Washington, 2004.