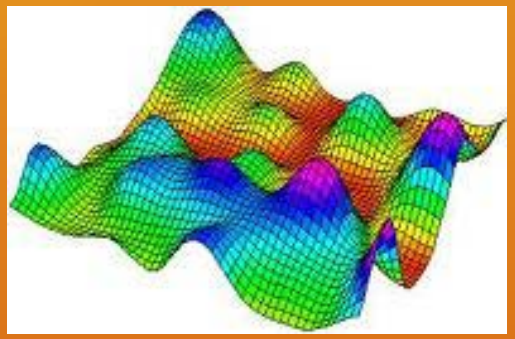
Un Algoritmo con Inspiración Biológica para Resolver Problemas de Optimización con Varios Objetivos

Dr. Carlos A. Coello Coello

Departamento de Computación

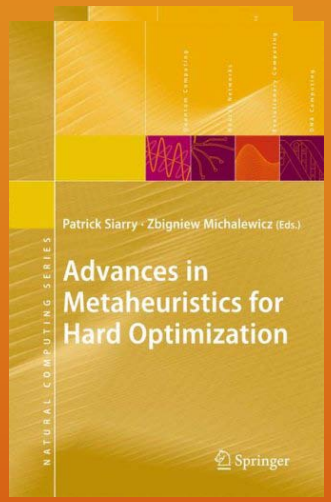
CINVESTAV-IPN

Conceptos Básicos



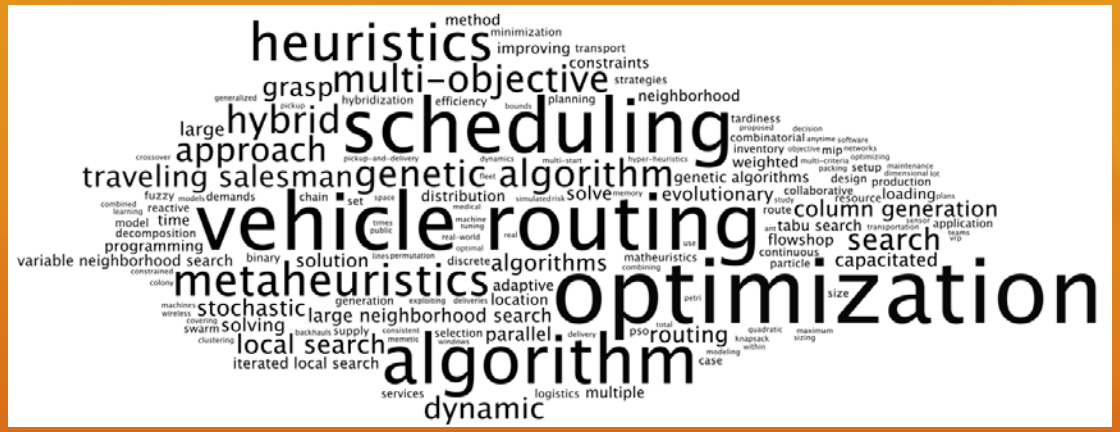
- Optimizar es encontrar la mejor solución posible a un problema, dado un cierto conjunto de condiciones. Existen problemas de optimización de diferentes grados de dificultad. En mi grupo de investigación, trabajamos fundamentalmente en torno al diseño y aplicación de metaheurísticas bio-inspiradas para resolver problemas de optimización de alto grado de dificultad.

Conceptos Básicos



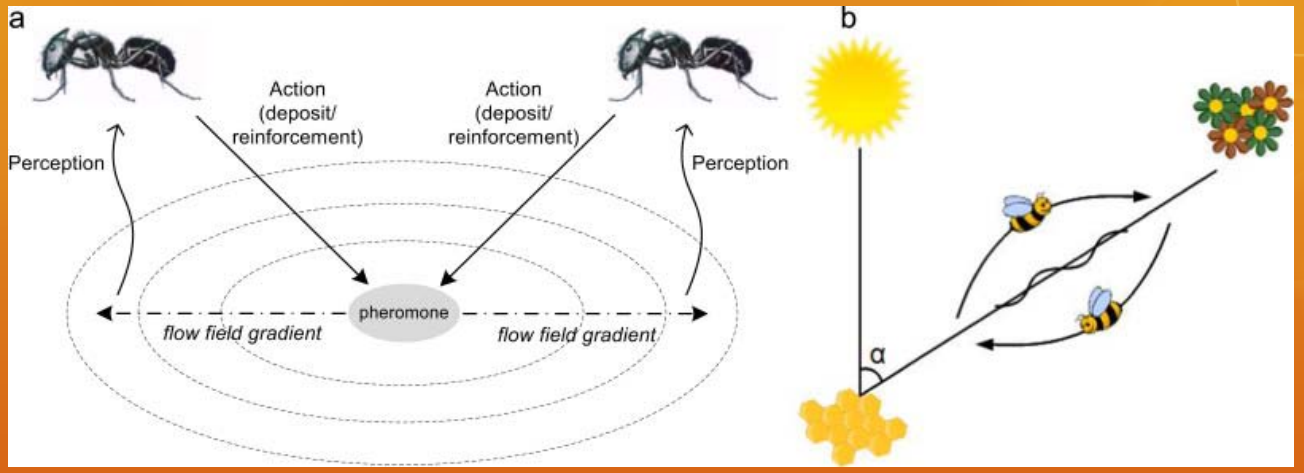
Las metaheurísticas son procedimientos de búsqueda de algo nivel que aplican ciertos tipos de reglas con base en alguna fuente de conocimiento, a fin de poder realizar una exploración más eficiente del espacio de búsqueda.

Conceptos Básicos



- Aunque las metaheurísticas no pueden garantizar (en general) que llegarán al óptimo de un problema, en la práctica suelen obtener soluciones razonablemente buenas en tiempos de cómputo razonablemente cortos. Debido a su flexibilidad y facilidad de uso, las metaheurísticas se han vuelto muy populares en los últimos 20 años, sobre todo cuando intentamos resolver problemas de alto nivel de complejidad.

Conceptos Básicos

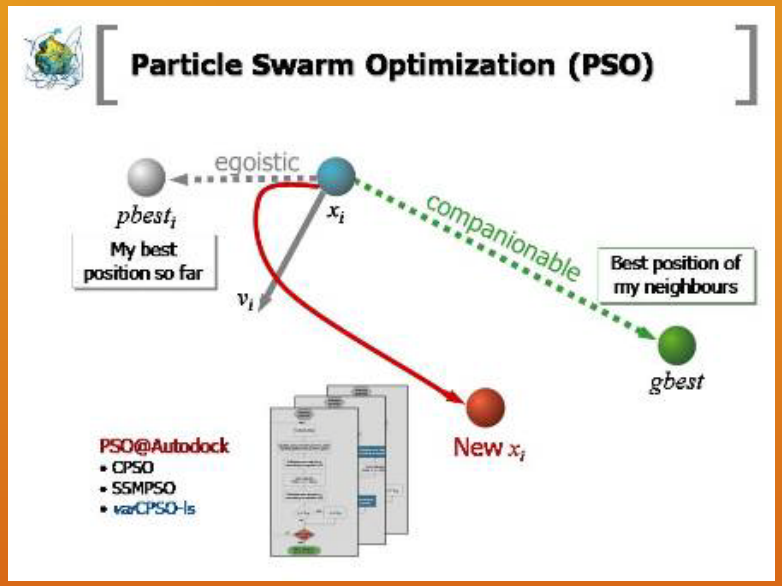


- Existe toda una familia de metaheurísticas cuyo diseño tiene una inspiración biológica, por lo cual se les denomina “metaheurísticas bio-inspiradas”. Algunas se basan en nuestro sistema inmune, otras en los patrones de vuelo de las aves, otras en la evolución de las especies, etc. Ejemplos de metaheurísticas bio-inspiradas son: el sistema inmune artificial, la colonia de hormigas, los algoritmos genéticos y la optimización mediante cúmulos de partículas.

Conceptos Básicos

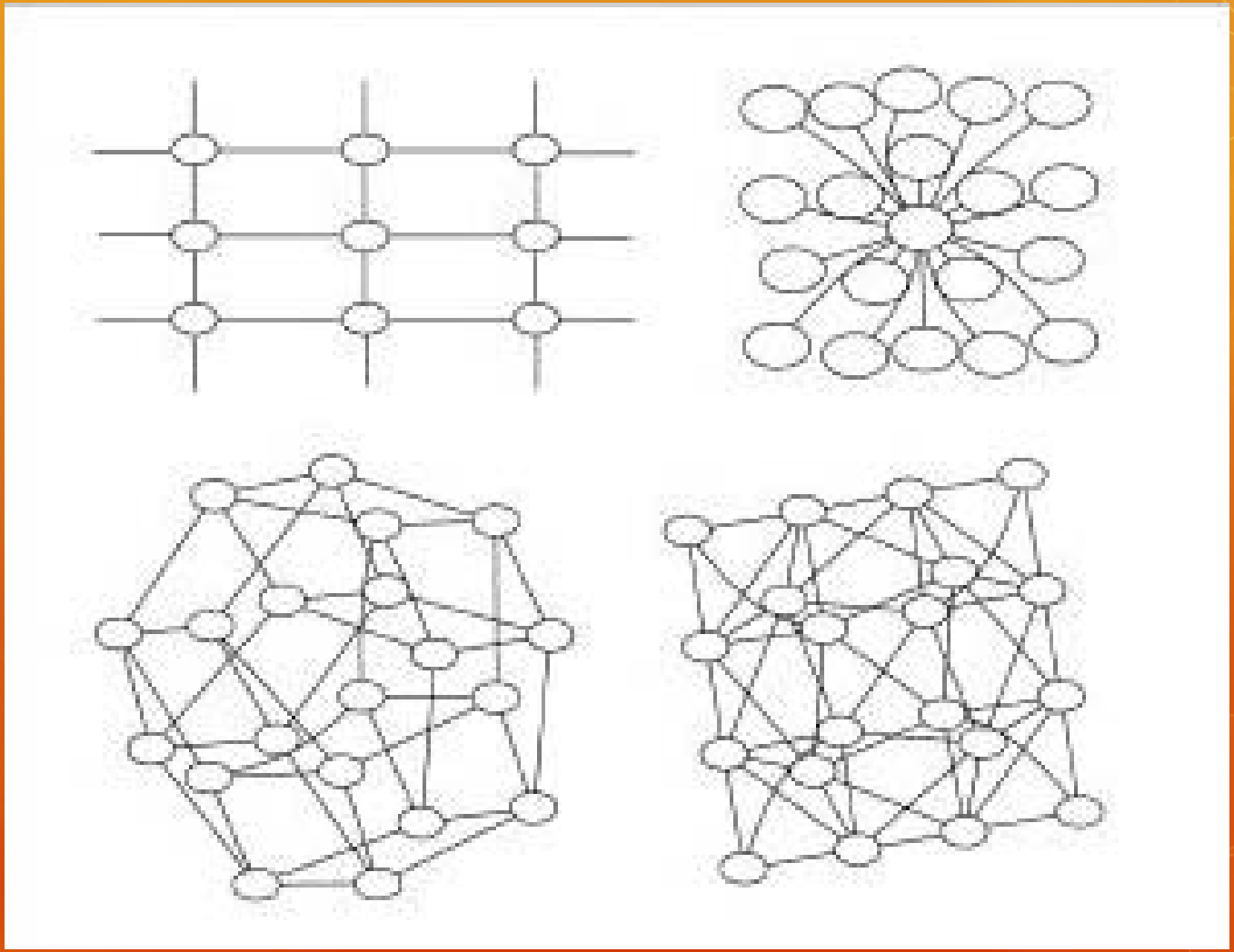
- En este caso, nuestro trabajo se centró en el uso de la metaheurística denominada “optimización mediante cúmulos de partículas”, la cual se basa en el patrón de vuelo de un conjunto de aves que salen en grupo a buscar comida, siguiendo a un líder. Este algoritmo se propuso en 1995 y ha sido muy popular para resolver problemas de optimización no lineal.

Conceptos Básicos



- En un optimizador mediante cúmulos de partículas (PSO, por sus siglas en inglés), se simula el movimiento de un conjunto de partículas en un cúmulo. Cada partícula se mueve con base en una velocidad que se actualiza a cada iteración. Para determinar su velocidad, una partícula recibe la influencia de la mejor posición que ha ocupado hasta el momento y de la mejor posición que han tenido todas las partículas hasta el momento. Además, se considera una inercia y dos factores adicionales (uno cognitivo y otro social), que afectan la velocidad de una partícula.

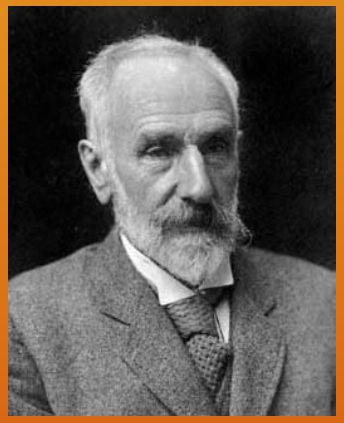
Conceptos Básicos

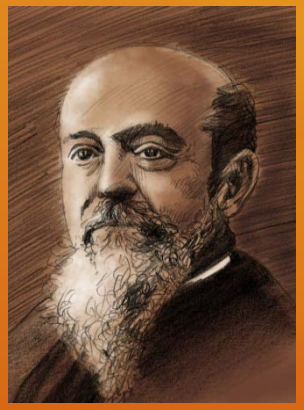


Optimización Multi-Objetivo

- La optimización multi-objetivo busca resolver problemas en los cuales queremos optimizar simultáneamente dos o más funciones objetivo, las cuales generalmente se encuentran en conflicto entre sí. Debido a su naturaleza, en estos problemas es necesario adoptar una nueva definición de “óptimo”. Este tipo de problemas son muy comunes en un gran número de disciplinas.

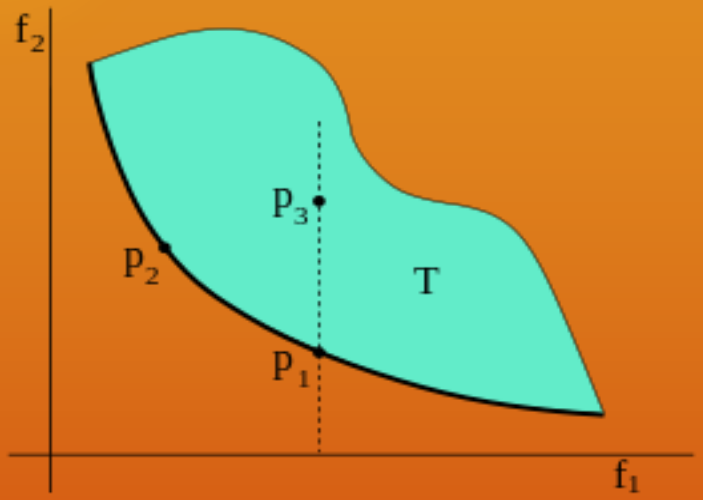
Optimización Multi-Objetivo





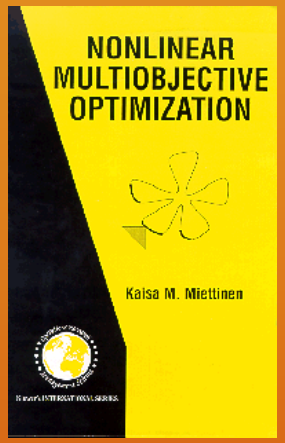
- La noción más comúnmente adoptada de “óptimo” en este contexto, fue propuesta originalmente por Francis Ysidro Edgeworth (en 1881) y fue posteriormente generalizada por el economista italiano Vilfedo Pareto (en 1896). A este concepto se le conoce como “optimalidad de Pareto” y se refiere a la obtención de “soluciones compromiso” en las cuales no es posible lograr una mejora de un objetivo sin perjudicar otro.

Optimización Multi-Objetivo



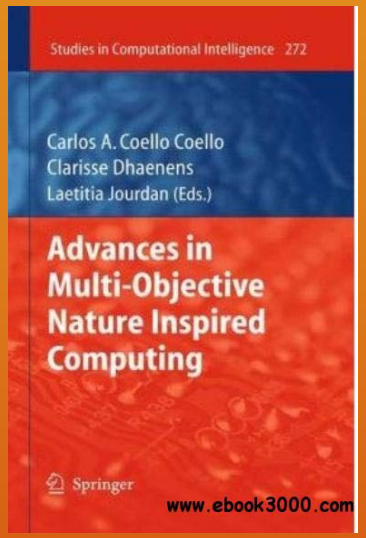
- La optimalidad de Pareto genera varias soluciones, que conforman el llamado “conjunto de óptimos de Pareto”. Los valores de las funciones objetivo correspondientes al conjunto de óptimos de Pareto forman el denominado “frente de Pareto”.

Optimización Multi-Objetivo



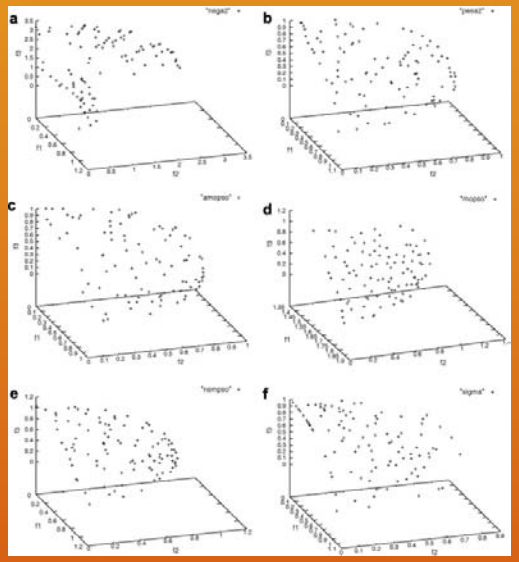
- Existen diversas técnicas de programación matemática para resolver problemas de optimización multi-objetivo. Sin embargo, dichas técnicas tienen varias limitantes. Por ejemplo, algunas son susceptibles a la continuidad del frente de Pareto, otras requieren derivadas, etc. En general este tipo de técnicas generan un solo elemento del conjunto de óptimos de Pareto por ejecución y son susceptibles al punto inicial de búsqueda.

Optimización Multi-Objetivo



- Las metaheurísticas suelen operar sobre un conjunto de soluciones, por lo cual, pueden generar varios elementos del conjunto de óptimos de Pareto en una sola ejecución, partiendo de un conjunto de soluciones aleatorias. Así mismo, son menos susceptibles a la forma y continuidad del frente de Pareto que las técnicas de programación matemática.

Optimización Multi-Objetivo



- El uso de metaheurísticas para resolver problemas multi-objetivo, es un área activa de investigación desde mediados de los 1980s. Sin embargo, el diseño de optimizadores multi-objetivo basados en cúmulos de partículas comenzó a finales de los 1990s.

Optimización Multi-Objetivo

- Nosotros incursionamos en el diseño de optimizadores multi-objetivo basados en cúmulos de partículas en 2002 (con la tesis de maestría de Maximino Salazar Lechuga). Nuestro algoritmo inicial fue presentado en un congreso internacional y fue uno de los primeros en incorporar la optimalidad de Pareto en el mecanismo de selección de líderes. Este trabajo llamó mucho la atención en su momento y a la fecha reporta unas 200 citas.

Sobre el Artículo


IEEE TRANSACTIONS ON
EVOLUTIONARY
COMPUTATION

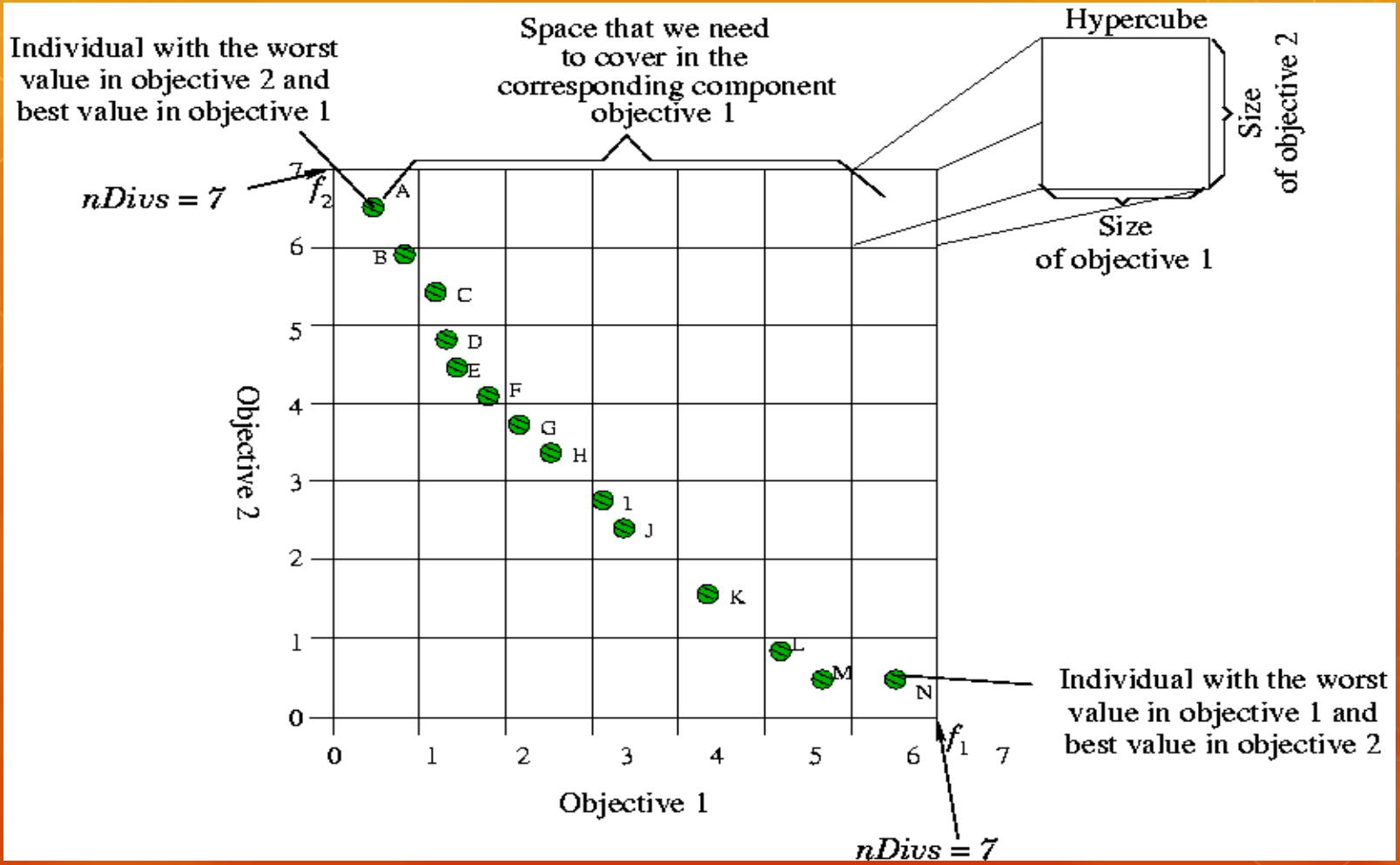
- El artículo en el que se enfoca esta plática presentó una versión revisada del algoritmo antes indicado (al cual llamamos MOPSO), con algunas mejoras, como un operador de mutación “inteligente”, una población secundaria con un sistema coordenado que permitía mantener diversidad y una validación mucho más exhaustiva que la realizada previamente a cualquier otro algoritmo similar. El artículo se sometió al *IEEE Transactions on Evolutionary Computation* en julio de 2002 y fue finalmente publicado en junio de 2004.

Aspectos Relevantes de MOPSO

- VEL[i] = W x VEL[i] + R1 x (PBESTS[i] – POP[i]) + R2 x (REP[h] – POP[i]);
- POP[i] = POP[i] + VEL[i];

- En estas ecuaciones, W (inercia) toma un valor de 0.4; R1 y R2 son números aleatorios en el rango [0..1]; PBESTS[i] es la mejor posición que la partícula “i” ha tenido hasta ahora; REP[h] es un valor tomado del repositorio (h es un índice que se genera con base en la diversidad del repositorio); POP[i] es el valor actual de la partícula “i”.

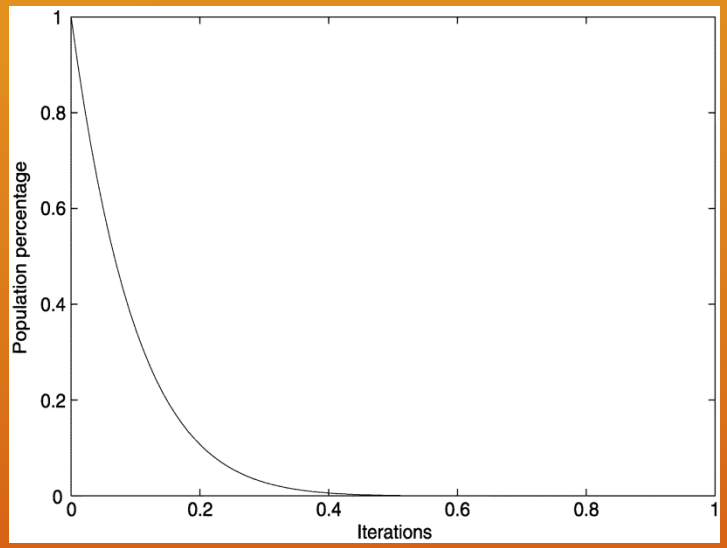
Aspectos Relevantes de MOPSO



Aspectos Relevantes de MOPSO

- Se diseñó un operador de mutación que intentaba mejorar la capacidad de exploración de nuestro MOPSO. La idea principal era forzar al algoritmo a explorar regiones del espacio de búsqueda que las fórmulas originales del PSO no lograban alcanzar. El operador tiene una aplicabilidad variable (se aplicaba con más intensidad al inicio de la búsqueda y su uso se iba diluyendo hacia el final). Este operador mejoró significativamente el desempeño de nuestro algoritmo. Este operador lo propuso Gregorio Toscano Pulido.

Aspectos Relevantes de MOPSO



- También incorporamos un mecanismo sencillo de manejo de restricciones: aplicamos una selección mediante torneo binario basado en factibilidad. Si los 2 individuos comparados eran factibles, se aplicaba la dominancia de Pareto para decidir al ganador. Si uno era factible y el otro no, el factible ganaba. Si los 2 eran infactibles, el que violaba en menor magnitud las restricciones del problema era el ganador. Este mecanismo ya lo habíamos usado antes en el microGA.

Validación

- Nuestro MOPSO se comparó contra 3 algoritmos representativos del estado del arte de aquella época:

- 1) **Nondominated Sorting Genetic Algorithm-II (NSGA-II):** Propuesto originalmente en 2000, se convirtió un estándar en la literatura. Principales fortalezas: gran eficacia, relativa sencillez del diseño algorítmico y buena velocidad.

- 2) **Pareto Archived Evolution Strategy (PAES):** Quizás sea el algoritmo evolutivo multi-objetivo más simple que pueda concebirse. Lo usamos porque la rejilla adaptativa de nuestro MOPSO es una variante de la de PAES.

- 3) **El Micro-Algoritmo Genético para Multi-Objetivo (microGA):** Diseñado por Gregorio Toscano. Un algoritmo que consume pocos recursos de cómputo y que produce buenos resultados.

Validación

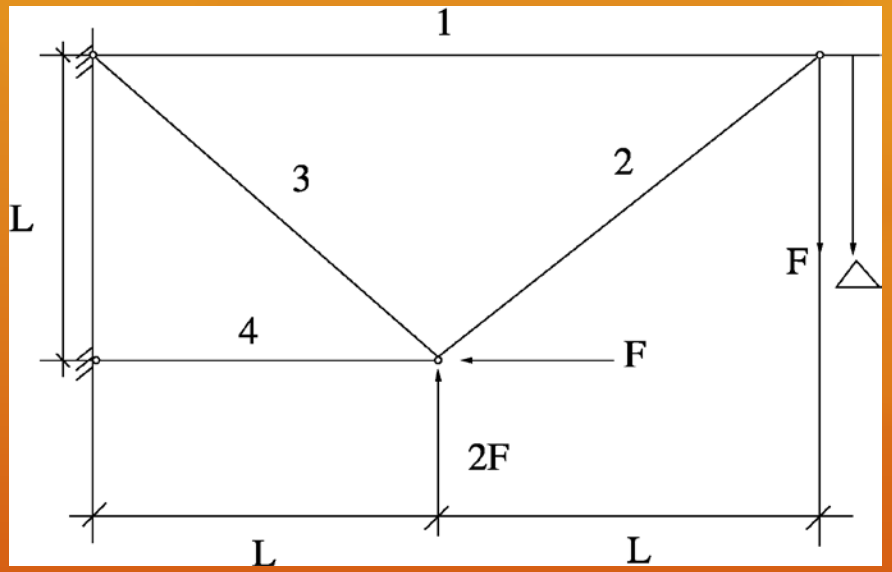
- Al medir el desempeño de un algoritmo evolutivo multi-objetivo, se busca satisfacer 3 metas:

- 1) Minimizar la distancia del frente obtenido con respecto al “verdadero” frente de Pareto del problema. Para medir esto, usamos la “distancia generacional”.

- 2) Maximizar la dispersión de soluciones a lo largo del frente y hacer que éstas tengan una distribución tan uniforme como sea posible. Para medir esto, usamos el “espaciado”.

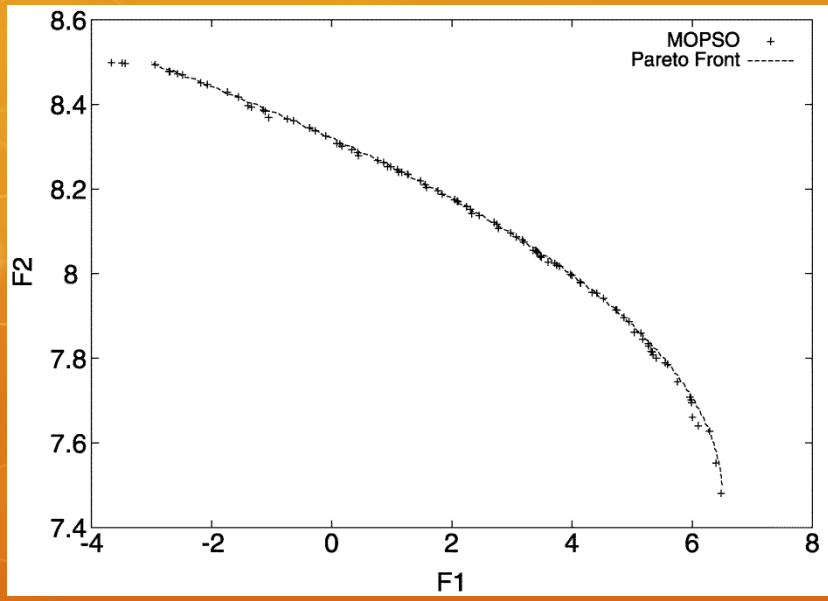
- 3) Maximizar la cantidad de elementos del conjunto de óptimos de Pareto obtenidos. Para medir esto usamos la “tasa de error”.

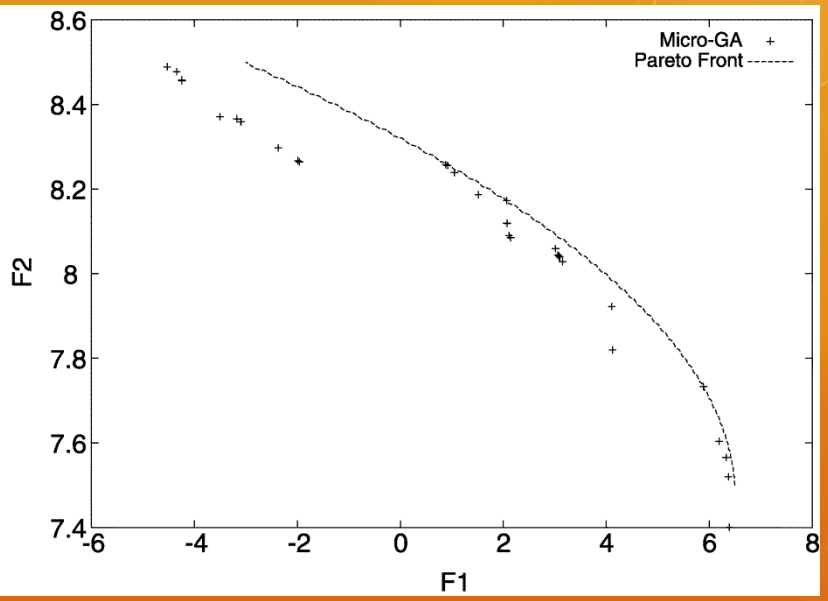
Validación

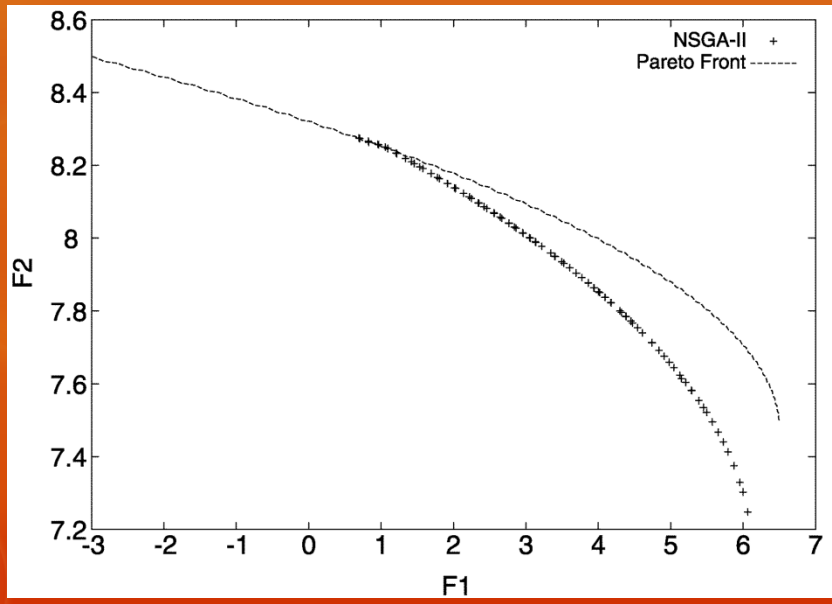


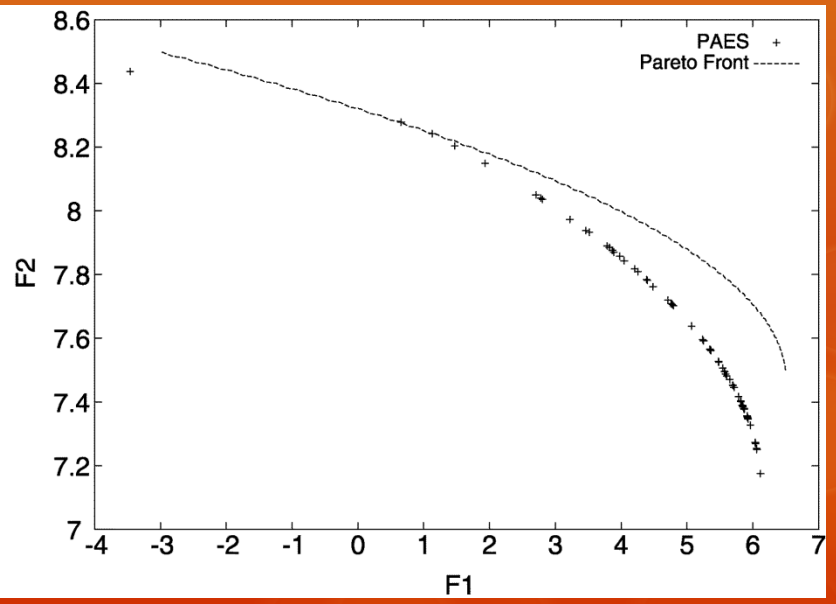
- Usamos 5 problemas de prueba, incluyendo dos ejemplos de optimización estructural. Se realizaron 30 ejecuciones independientes de cada algoritmo para cada problema, usando parámetros para cada algoritmo que permitiera una comparación justa.

Resultados

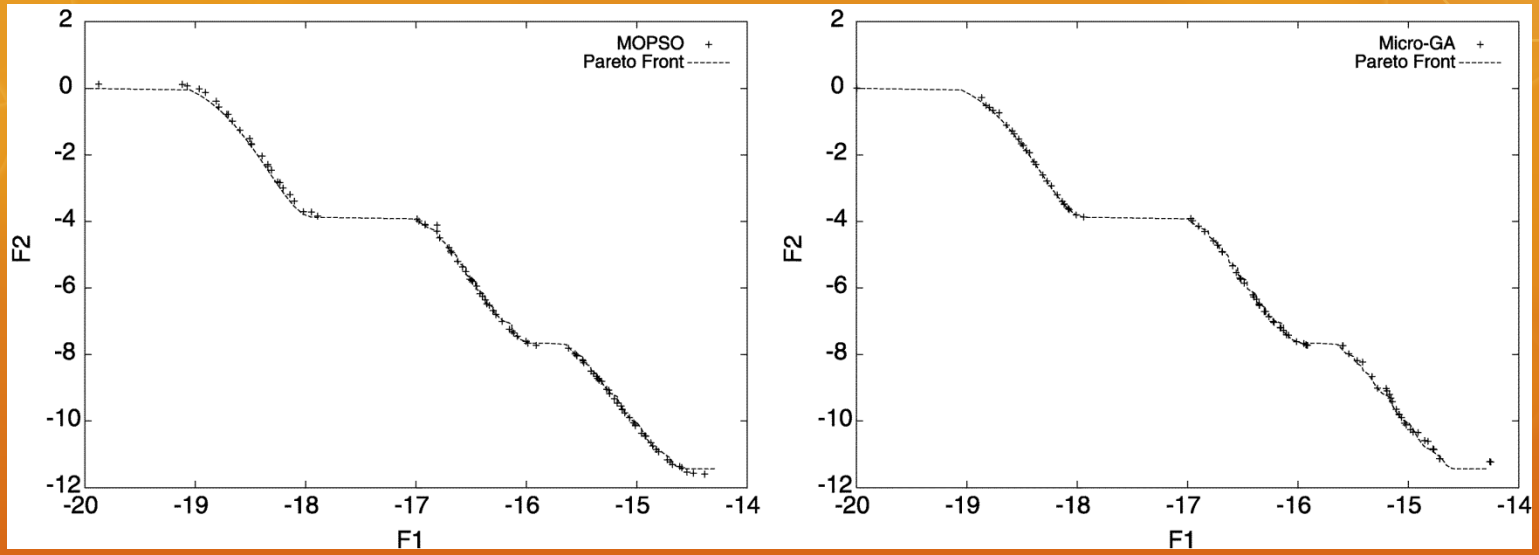

MOPSO
Pareto Front
F1
F2


Micro-GA
Pareto Front
F1
F2

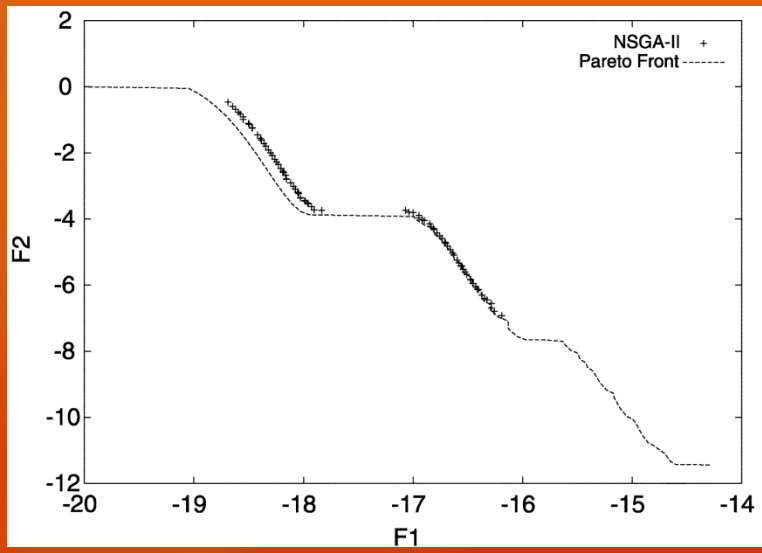

NSGA-II
Pareto Front
F1
F2

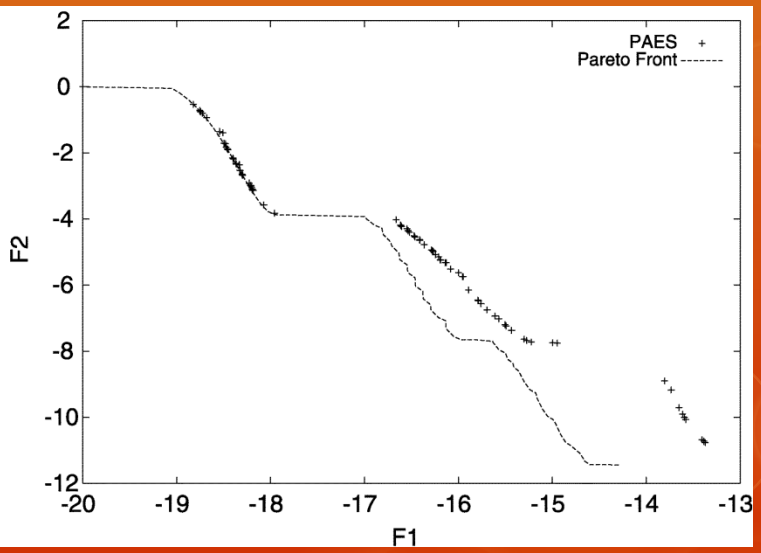

PAES
Pareto Front
F1
F2

Resultados


MOPSO
Pareto Front
F1
F2
2
0
-2
-4
-6
-8
-10
-12
-20
-19
-18
-17
-16
-15
-14

Micro-GA
Pareto Front
F1
F2
2
0
-2
-4
-6
-8
-10
-12
-20
-19
-18
-17
-16
-15
-14


NSGA-II
Pareto Front
F1
F2
2
0
-2
-4
-6
-8
-10
-12
-20
-19
-18
-17
-16
-15
-14


PAES
Pareto Front
F1
F2
2
0
-2
-4
-6
-8
-10
-12
-20
-19
-18
-17
-16
-15
-14
-13

Resultados

time	MOPSO	NSGA-II	microGA	PAES
Best	0.155	2.181	0.295	0.938
Worst	0.168	2.693	0.345	1.39
Average	**0.162158**	2.426789	0.32695	1.12615
Median	0.161	2.461	0.3325	1.121
Std. Dev.	0.003468	0.171008	0.014813	0.105224

- En general, nuestro MOPSO obtuvo los mejores resultados con respecto a la distancia generacional (convergencia), y quedó en segundo lugar (después del NSGA-II) con respecto a la dispersión. Adicionalmente, nuestro algoritmo resultó bastante más rápido que el NSGA-II (en algunos casos, fue hasta 15 veces más rápido, en términos de tiempo de CPU).

Análisis de Sensibilidad

- Se realizó un análisis de sensibilidad para detectar qué parámetros eran los más adecuados para nuestro MOPSO. De él se concluyó lo siguiente:

- 1) Tamaño de población: Se recomienda usar 100 partículas.
- 2) Número de subdivisiones de la rejilla: Se recomienda usar al menos 30.
- 3) Tamaño del repositorio: Se recomienda usar 250 partículas.
- 4) Mutación: Se recomienda su uso.
- 5) Iteraciones: Se recomienda usar al menos 100.

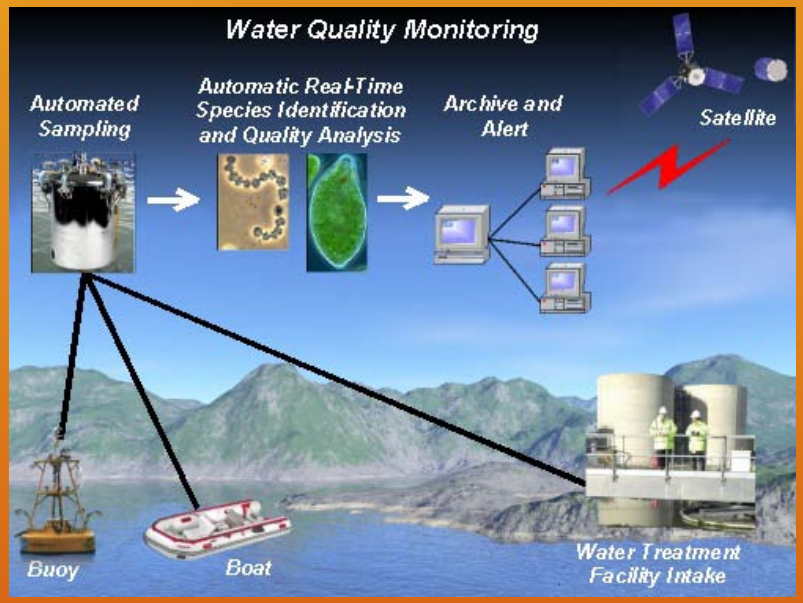
Puntos a destacar

- MOPSO es un algoritmo relativamente simple y muy rápido (resultó incluso más rápido que el microGA en algunos casos) y logró superar significativamente (en convergencia) a PAES y al microGA, ganándole marginalmente al NSGA-II, pero con mucho menores tiempos de cómputo.
- Un aspecto en el cual el algoritmo debía mejorarse era el de su distribución de soluciones.
- El código fuente del algoritmo ha estado disponible desde la publicación del artículo.

Puntos a destacar

- Este algoritmo inició una línea de investigación en torno al diseño de MOPSOs de segunda generación, más poderosos y sofisticados que los desarrollados a fines de los 1990s. Diversos grupos de investigación alrededor del mundo han trabajado en esta área desde entonces.
- Este algoritmo es discutido a detalle en un libro titulado "Fundamentals of Computational Swarm Intelligence" escrito por Andries Engelbrecht (Wiley, 2005).
- Este artículo ha sido citado más de 360 veces (unas 245 de estas citas están registradas en el Citation Index). A la fecha, sigue siendo uno de los artículos más citados del *IEEE Transactions on Evolutionary Computation* y uno de los más citados en computación de los últimos 10 años.

Puntos a destacar



- Diversos autores de algoritmos evolutivos multi-objetivo han comparado resultados con respecto a nuestro MOPSO. Así mismo, se han desarrollado variantes de nuestro algoritmo que se han aplicado a la solución de problemas del mundo real (p.ej., Baltar y Fontane (2006) lo usaron para resolver problemas de la calidad del agua).

A manera de conclusiones

Timing is everything…

- El diseño de este algoritmo comenzó como una mera curiosidad, debido al buen desempeño de PSO en problemas mono-objetivo.
- A la larga, sin embargo, se convirtió en una línea de investigación muy prometedora, en la cual hemos seguido trabajando desde entonces (hemos diseñado algoritmos como EMOPSO, OMOPSO, microPSO, etc.). De hecho, Gregorio Toscano Pulido acabó por escribir su tesis doctoral en torno a un MOPSO y realizó el primer estudio exhaustivo de sensibilidad de este tipo de algoritmos.