Optimización Multiobjetivo Usando un Micro Algoritmo Genético.

por

Gregorio Toscano Pulido

Para obtener el grado de Maestro en Inteligencia Artificial.

Maestría en Inteligencia Artificial
Universidad Veracruzana - LANIA

Dr. Carlos A. Coello Coello
CINVESTAV-IPN
Asesor

Dr. Carlos E. Mariano Romero
Instituto Mexicano de Tecnología del Agua
Revisor

Dra. Katya Rodriguez Vázquez
IIMAS-UNAM
Revisor

Septiembre de 2001

Resumen

Los problemas del mundo real, en su gran mayoría presentan diferentes objetivos a optimizarse (los cuales generalmente se expresan en unidades diferentes y están en conflicto entre sí) y un espacio de búsqueda grande y complejo (p.ej., no diferenciable, accidentado, etc.). Estas dos características son suficientes para que los métodos tradicionales de optimización resulten inoperantes o simplemente requieran un costo computacional prohibitivo. Por lo tanto, este tipo de problemas requieren de técnicas alternativas de solución. La computación evolutiva ha sido utilizada satisfactoriamente para resolver los denominados problemas multiobjetivo (o sea, con más de una función objetivo). Sin embargo, no fue sino hasta recientemente que se ha comenzado a enfatizar el desarrollo de algoritmos que sean no sólo efectivos, sino también eficientes (en términos computacionales). Como resultado de estos estudios, esta disciplina conocida como optimización evolutiva multiobjetivo, ha originado nuevas metodologías para el desarrollo de algoritmos eficientes. Se sabe que las dos fuentes principales de ineficiencia de un algoritmo evolutivo multiobjetivo son: (a) el proceso de jerarquización para clasificar a los individuos usando el concepto de dominancia de Pareto y (b) el mecanismo utilizado para preservar la diversidad en la población. Con esto en mente, se han propuesto recientemente algunas formas en que resulta posible disminuir los costos asociados con estas dos operaciones.

El objetivo principal de esta tesis fue el de construir un algoritmo eficiente, basado en tendencias novedosas del área que permiten el desarrollo de técnicas eficientes para solventar problemas de optimización multiobjetivo. Como consecuencia, se decidió desarrollar una aplicación en la cual se usaría un algoritmo genético con una población de tamaño muy pequeño y un proceso de reinicialización combinado con un archivo externo para guardar los vectores no-dominados encontrados previamente y un mecanismo para mantener diversidad basado en distribución geográfica, para resolver problemas de optimización multiobjetivo de diferentes grados de complejidad. Para validar el desempeño del algoritmo propuesto, se evaluó su comportamiento en seis funciones de prueba tomadas de la literatura especializada y se compararon los resultados obtenidos contra los producidos por dos técnicas recientes.

Dedicatorias

Esta tesis esta especialmente dedicada a aquellas dos especiales personas que originaron mi vida y que la han llenado de amor, orgullo, esperanza y deseos de vivir: a mi aliento, mi madre y a mi héroe, mi padre.

También quiero dedicarla a mis hermanas y abuela por todo el apoyo y fé que siempre han puesto en mí.

Agradecimientos

- La realización de esta tesis marca la culminación de otra etapa de mi vida. Sin embargo, ésta no habría podido llevarse a cabo sin la intervención, apoyo y guía de un incontable número de personas. Injustamente, no puedo mencionar el nombre de cada uno de ellos. Sin embargo, les doy mis más sinceras gracias.

  Quiero agradecer a mi familia: tíos, primos, abuelas, hermanas y padres, por brindarme su apoyo incondicional en cada momento de mi existencia.

  Quiero agradecer a todos aquellos profesores que han hecho que el estudio valga la pena para mi. En especial, a mi maestro de matemáticas de la secundaria, Armando Gutiérrez, por ser, ademas, un amigo y guía en esa etapa especial de mi vida.

  Agradezco a todas esas personas maravillosas que Dios ha puesto en mi camino y quienes con gran orgullo llamo amigos.

  Agradezco a mi asesor, el Dr. Carlos Coello Coello por extralimitar sus funciones de guía, llegando a ser un gran amigo, en quién puedo confiar.

  Agradezco el apoyo otorgado por CONACyT a través de una beca con registro 141617 para cursar la Maestría en Inteligencia Artificial de la Universidad Veracruzana. Asimismo, agradezco el apoyo recibido a través del proyecto CONACyT 34201-A, titulado ”Nuevos Paradigmas en Optimización Evolutiva Multiobjetivo”, cuyo responsable es el Dr. Carlos A. Coello Coello.

Índice General

Índice de Figuras

Índice de Tablas

Introducción

Lidiar con problemas multiobjetivo es un problema cotidiano que se presenta hasta en los más mínimos procesos, tanto personales como industriales. El hecho de que no sea un problema reciente, ha permitido que se desarrollen diversas técnicas eficientes. Sin embargo, la gran mayoría de los problemas del mundo real, tienen un espacio de búsqueda muy grande en las que técnicas tradicionales no son aptas para ser utilizadas, por lo que es necesario recurrir a técnicas alternativas. La computación evolutiva, es un conjunto de algoritmos que presentan una serie de características idóneas para resolver este tipo de problemas. Sin embargo, a pesar de que ha estado presente en la resolución de problemas de optimización multiobjetivo desde mediados de los 80s, no fue sino hasta recientemente que se ha comenzado a realizar investigación sobre la eficiencia de sus algoritmos. Tener algoritmos eficientes, se refleja directamente en la calidad de las soluciones que éstos pueden encontrar y en la velocidad de ejecución que dura cada corrida.

En esta tesis se muestra cómo un algoritmo genético cuidadosamente diseñado es suficiente para generar el frente de Pareto de problemas de optimización multiobjetivo complejos. Dicha aplicación no solamente reduce la cantidad de comparaciones para generar el frente de Pareto (con respecto a tradicionales EMOO aplicaciones basadas en jerarquización de Pareto), sino que reduce drásticamente el tiempo-máquina de ejecución por corrida.

La organización de ésta tesis es la siguiente: **Capítulo 1** Provee una revisión histórica de la computación evolutiva, abarcando desde el origen de sus bases científicas y concluyendo con una breve descripción de sus principales paradigmas. **Capítulo 2** contiene las definiciones y conceptos básicos necesarios para comprender la problemática de la optimización multiobjetivo, enumera las principales técnicas de programación matemática y nos muestra el estado del arte de los algoritmos genéticos para problemas de optimización multiobjetivo. **Capítulo 3** brinda una descripción detallada de la técnica propuesta, las capacidades y limitaciones de cada componente y una descripción de sus parámetros. **Capítulo 4** describe las métricas utilizadas, las funciones de prueba y los resultados obtenidos por el algoritmo propuesto y por las técnicas con quienes se le compara. Finalmente se presentan las conclusiones obtenidas de la elaboración de éste trabajo y algunas rutas futuras de investigación.

Capítulo 1

Computación evolutiva

1.1 Introducción

El término Computación Evolutiva[1] *(CE)*, engloba a procesos de adaptación inspirados en los mecanismos de la evolución y selección natural (Neo-Darwinismo), y utilizando operadores genéticos mantienen una población de individuos en constante evolución. La CE tiene un domino de aplicación muy extenso, la cual va desde la optimización numérica de una variable hasta la optimización multiobjetivo pudiendo involucrar diferentes variables en el proceso.

La CE, tuvo como primer inspirador al biólogo Alexander S. Fraser, quién desarrolló trabajos a fines de los 50s acerca de la evolución de sistemas biológicos.A raíz de estos trabajos las aportaciones de científicos tales como Box, Friedberg, Friedman, Barricelli y Bremermann, se sumaron rápidamente, quedando sentadas las bases de la misma [23].

En la actualidad la CE está inscrita dentro de la inteligencia artificial (IA) como un claro ejemplo de su escuela subsimbólica y está circunscrita dentro del aprendizaje máquina no supervisado y la búsqueda heurística.

En este capítulo se proporcionan los antecedentes históricos que conllevaron a la formulación del Neo-Darwinismo, así como una breve discusión sobre la computación evolutiva y sus principales paradigmas.

1.2 Antecedentes conceptuales de la Genética

La genética es una rama del conocimiento muy joven, y aunque sus comienzos datan del 1000 A.C. no fue sino a hasta 1930 que se afianzó como ciencia mediante la teoría sintética de la evolución.

Antes que esto sucediera se tuvieron que producir y madurar los avances conceptuales descritos a continuación.

[1] Conocida también como Algoritmos Evolutivos o Técnicas Evolutivas.

1.2.1 Adaptacionismo, Jean Baptiste de Lamarck (1800)

La filosofía zoológica de *Lamarck (1744-1829)* hizo época en los anales de la ciencia al presentar su teoría de la evolución [31], expuesta en *Philosophie Zoologique*, a pesar que sus errores de concepto fueran hechos evidentes más tarde.

Su hipótesis se basa en el siguiente razonamiento:

> Una gran modificación en el ambiente de una especie animal daría por resultado la necesidad de un cambio en esa determinada especie. Esta necesidad de cambio, producida por sentimientos internos, causaría en el animal la formación de nuevos hábitos para ajustarse a su nuevo ambiente. Esto dio origen a las siguientes dos suposiciones:

Ley del uso y del desuso En todo animal que no haya superado el término de su desarrollo, el empleo más frecuente y sostenido de un órgano cualquiera, fortifica poco a poco este órgano, lo desarrolla, lo aumenta y le da una fuerza proporcional a la duración de este empleo; mientras que el defecto constante de uso de tal órgano lo debilita insensiblemente, lo deteriora, disminuye progresivamente sus facultades y termina por hacerlo desaparecer.

Ley de la herencia de los caracteres adquiridos Todo lo que la naturaleza ha hecho adquirir o perder en los individuos por la influencia de las circunstancias en donde su raza se encuentra expuesta desde hace mucho tiempo y en consecuencia por la influencia del empleo predominante de tal órgano, o por la de un defecto constante de uso de tal parte, la naturaleza lo conserva por la generación en los nuevos individuos que provienen de allí, con tal que los cambios adquiridos sean comunes a ambos sexos, o en los que han favorecido a los nuevos individuos.

Mediante estas suposiciones, Lamarck explicó el mecanismo de la evolución, asegurando que las especies evolucionaban después de muchas generaciones como resultado de la adquisición o pérdida de caracteres.

1.2.2 Selección Natural, Charles Robert Darwin (1859)

El naturalista británico *Charles Darwin (1809-1882)* introduce en su libro *El origen de las especies* [15] uno de los avances conceptuales más importantes del siglo XIX, al haber sentado las bases de la teoría moderna de la evolución mediante el concepto del desarrollo de todas las formas de vida a través del proceso lento de la selección natural. Su trabajo repercutió decisivamente sobre las ciencias que corresponden a la vida y a la tierra, y sobre el pensamiento moderno.

La teoría de la evolución biológica reza que las formas orgánicas ahora existentes proceden de otras distintas que existieron en el pasado, mediante un proceso de descendencia con modificación, siendo la selección el proceso natural más importante que hace posible dicha modificación.

La selección natural sostiene que a causa del problema de la disponibilidad de alimentos, los jóvenes miembros de las distintas especies compiten intensamente por su supervivencia. Los que sobreviven, que darán lugar a la siguiente generación, tienden a incorporar variaciones naturales favorables al proceso de selección natural, y estas variaciones se transmitirán a través de la herencia.

1.2.3 Trasmisión de los caracteres hereditarios, Gregor Mendel (1865)

El científico austriaco *Gregor Johann Mendel (1822 - 1884)* fue pionero en el estudio sobre la transmisión de los caracteres hereditarios. En 1865 presentó su trabajo a la Sociedad de Historia Natural de Brün, con el título *"Híbridos en plantas"* [45]. En él se resumían experimentos que había llevado a cabo durante 8 años usando chícharos *(Pisum sativum)* y explica cómo van a ser las características de los descendientes, a partir del conocimiento de las características de los progenitores.

Sus experimentos son el paradigma del análisis genético y su trabajo es considerado fundamental en la genética. Un diseño experimental sencillo junto con un análisis cuantitativo de sus datos fue la fuerza principal de su trabajo.

Los experimentos demostraron que:

a) la herencia se transmite por elementos particulados (refutando, por tanto, la herencia de las mezclas) y que

b) siguen normas estadísticas sencillas.

Con la anterior conclusión Mendel desarrolló las siguientes 3 leyes:

Ley de la uniformidad Establece que si se cruzan dos razas puras para un determinado carácter, los descendientes de la primera generación son todos iguales entre sí e iguales a uno de los progenitores.

Ley de la disyunción Establece que los caracteres recesivos, al cruzar dos razas puras, quedan ocultos en la primera generación, reaparecen en la segunda en proporción de uno a tres respecto a los caracteres dominantes.

Ley de la segregación independiente Establece que los caracteres son independientes y se combinan al azar.

1.2.4 Teoría del Plasma Germinal, August Weismann (1883)

El alemán *August Weismann (1834-1914)* enuncia en 1883 la teoría de que las partículas hereditarias (biófora) eran invisibles, autorreplicativas y asociadas con los cromosomas de un modo lineal y postuló que cada biófora estaba implicada en la determinación de una característica. Esta teoría fue denominada *continuidad del plasma germinal* [69]. En ella se reconocen dos tipos de tejidos en los organismos, el somatoplasma y el germoplasma. El primero forma la mayor parte del cuerpo de un individuo, mientras que el germoplasma era una porción inmortal de un organismo que tenía la potencialidad de duplicar a un individuo. A diferencia de la teoría de la pangénesis, el germoplasma no proviene del somatoplasma ni se forma nuevamente en cada generación, sino que constituye la continuidad de la información genética entre generaciones. Su teoría rechazaba rotundamente la herencia de los caracteres adquiridos y supuso un mayor énfasis en el material hereditario.

1.2.5 Neo-Darwinismo

Se llamó *Neo-Darwinismo* a la fusión de la teoría de la evolución por selección natural, las leyes de la herencia de Mendel y la hipótesis del plasma germinal de Weismann.

El pensamiento evolutivo actual gira en torno al *Neo-Darwinismo*, el cual establece que toda la vida en el planeta puede ser explicada a través de:

- Reproducción: Mediante este mecanismo se asegura que el material genético se herede de una generación a otra.
- Mutación: Se produce cuando ocurre un error en la copia del material genético durante la reproducción. Se dice que algunas mutaciones pueden resultar benéficas para los organismos. Si la mutación es benéfica, entonces, el organismo se adapta mejor a su medio.
- Competencia: Es un proceso natural, en la que los organismos tienen una lucha continua por sobrevivir y heredar su código genético a otra generación.
- Selección: Los organismos más aptos, tendrán más oportunidad de sobrevivir y por ende de reproducirse.

1.3 Computación evolutiva

La computación evolutiva es capaz de ir generando soluciones para problemas del mundo real gracias a que trabaja con base análoga al Neo-Darwinismo (particularmente el hecho de que las poblaciones a lo largo de las generaciones evolucionan en la naturaleza acorde a los principios de la selección natural y la supervivencia de los más aptos). La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de que se implementen adecuadamente los siguientes puntos:

- Codificar las estructuras que se replicarán (es decir, representar adecuadamente las soluciones al problema a resolverse).
- Operaciones que incidan sobre los ”individuos”.
- Una función de aptitud, que juegue el papel del ambiente.
- Un mecanismo de selección.

En la actualidad la computación evolutiva está claramente afianzada dentro de las ciencias computacionales principalmente debido a ciertas características que denota, de las cuales destacan las siguientes:

- No necesitan conocimientos específicos sobre el problema que intentan resolver, resolviendo inclusive problemas para los cuales no se conoce solución alguna.
- Son conceptualmente muy simples y tienen amplia aplicabilidad.
- Pueden explotar fácilmente las arquitecturas en paralelo, en caso de estar disponibles.

Aunadas a éstas, también se encuentran diferentes características de las técnicas evolutivas (TE) con respecto a las tradicionales: Las TE usan una población de soluciones potenciales en vez de un solo individuo, lo cual las hace menos susceptibles a quedar atrapadas en mínimos/máximos locales; usan operadores probabilísticos, mientras las técnicas tradicionales utilizan operadores determinísticos, aunque este hecho no significa que operen de manera análoga a una simple búsqueda aleatoria; tienen el potencial para incorporar conocimiento sobre el dominio y para hibridizarse con otras técnicas de búsqueda/optimización; son robustas a los cambios dinámicos y generalmente pueden auto-adaptar sus parámetros.

Por la forma en como trabaja, la computación evolutiva se puede agrupar en tres paradigmas principales

- Programación evolutiva.
- Estrategias evolutivas.
- Algoritmos genéticos.

Adicionalmente es importante mencionar una variante del algoritmo genético que ha adquirido gran popularidad en los últimos años: la programación genética.

1.4 Programación evolutiva

La Programación Evolutiva (PE), se fundamenta en las ideas expresadas por *Lawrence J. Fogel* quien concibió el uso de la evolución simulada en la solución de problemas hacia mediados de los 60s [22]. En esta técnica la inteligencia se ve como un comportamiento adaptativo y se enfatiza los nexos de comportamiento entre padres e hijos.

La *PE* es una abstracción de la evolución al nivel de las especies, por lo que no se requiere el uso de un operador de recombinación (diferentes especies no se pueden cruzar entre sí). Asimismo, usa selección probabilística (generalmente por torneo estocástico). En la actualidad existen variantes de esta técnica con diversos dominios de aplicación.

El algoritmo básico de la programación evolutiva se muestra en la figura 1.1:

```
begin
  generar aleatoriamente la población inicial
  repeat
    aplicar Mutación
    calcular la aptitud de cada hijo
    seleccionar las soluciones que se retendrán
    reemplazar la población actual por la seleccionada
  until que cierta condición se satisfaga
end
```

Figura 1.1: Algoritmo general de la programación evolutiva.

1.5 Estrategias evolutivas

Hacia mediados de los 60s *Peter Bienert* [7], *Ingo Rechenberg* [53] y *Hans-Paul Schwefel* [60] desarrollaron un método de ajustes discretos aleatorios inspirado en el mecanismo de mutación que ocurre en la naturaleza. La técnica creada fue denominada *"estrategias evolutivas" (EE)* y fueron utilizadas inicialmente para resolver problemas hidrodinámicos de alto grado de complejidad.

La versión original $(1+1)-EE$ usaba un solo padre y con él se generaba un solo hijo. Este hijo se mantenía si era mejor que el padre, o de lo contrario se eliminaba (selección extintiva).

En la $(1+1)-EE$, un individuo nuevo es generado usando:

$$\overline{x}^{t+1} = \overline{x}^{t} + N(0,\sigma)$$

donde t se refiere a la generación (o iteración) en la que nos encontramos, y $N(0,\sigma)$ es un vector de números Gaussianos independientes con una media de cero y desviaciones estándar σ.

En la tabla 1.1 se muestran las aportaciones innovadoras que se le añadieron a la versión original $(1+1)-EE$.

En las estrategias evolutivas no sólo se evolucionan las variables del problema sino también los parámetros mismos de la técnica (desviación estándar), permitiendo con esto la auto-adaptación. La recombinación es posible gracias a que en la PE se trabaja a nivel de los individuos (fenotípico), la recombinación puede ser tanto sexual como panmíticas (cualquier individuo puede reproducirse con otro con una probabilidad que depende solamente de su aptitud) y la selección utilizada es generalmente determinista. Su algoritmo general se muestra en el pseudocódigo mostrado en la figura 1.2.

Aportación	Autor	Notación
Concepto de población μ padres y genera un solo hijo reemplazando al peor padre de la población.	Rechenberg (1973)	$(\mu+1)-EE$
Concepto de múltiples hijos a) Los μ mejores individuos obtenidos sobreviven. b) Sólo μ individuos obtenidos sobreviven.	Schwefel (1975)	a) $(\mu+\lambda)-EE$ y b) $(\mu,\lambda)-EE$
Regla del éxito $1/5$	Rechenberg	
Regla del éxito $1/7$	Thomas Bäck (1996)	

Tabla 1.1: Tabla de los diferentes tipos de estrategias evolutivas existentes

1.6 Algoritmos genéticos

Los algoritmos genéticos (denominados originalmente *planes reproductivos*) fueron introducidos por *John H. Holland* [29] a principios de los 60s y son, por mucho, la técnica evolutiva más popular de la actualidad. El objetivo de Holland fue el estudio formal de los procesos de adaptación natural y el traslado de estos mecanismos a la computación.

Una definición bastante completa de algoritmos genéticos es la brindada por Koza [40]:

> Es un algoritmo matemático altamente paralelo que transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo al principio Darwiniano de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual. Cada uno de estos objetos matemáticos suele ser una cadena de caracteres (letras o números) de longitud fija que se ajusta el modelo de las cadenas de cromosomas, y se les asocia con una cierta función matemática que refleja su aptitud.

Para poder aplicar el algoritmo genético se requiere de los siguientes seis componentes básicos:

1. Una *representación* de las soluciones potenciales del problema. La representación es la cadena binaria para la codificación de las soluciones del problema (a la cadena se le llama *cromosoma*). A cada posición de la cadena se le denomina *gene* y al valor dentro de esta posición se le llama *alelo*.

2. Una forma de crear una población inicial de posibles soluciones (normalmente un proceso aleatorio).

begin
  generar `aleatoriamente la población inicial`, $G(0)$
  sea $t := 0$
  evaluar $G(t)$
  repeat
    sea $t := t + 1$
    seleccionar $G_1(t)$ `de` $G(t)$
    recombinar $G_1(t)$ `para generar` $G(t+1)$
    mutar
    evaluar
    supervivientes
  until `que cierta condición se satisfaga`
end

Figura 1.2: Algoritmo general de las estrategias evolutivas.

3. Una *función de evaluación* que juegue el papel del ambiente, clasificando las soluciones en términos de su *aptitud*.

4. Un mecanismo de *selección* que permita seleccionar a los individuos de acuerdo a su aptitud. En los algoritmos genéticos se puede llevar el proceso de selección de diversas maneras, ya sea determinística o probabilísticamente (dando la oportunidad a los menos aptos para reproducirse). Las técnicas de selección se pueden agrupar en tres grupos:

   (a) Selección proporcional: Originalmente propuesta por *Holland* [29]. En esta selección los individuos se eligen de acuerdo a su aptitud con respecto a la población. Se subdivide en :

      i. Ruleta.
      ii. Sobrante estocástico.
      iii. Universal estocástico.
      iv. Muestreo determinístico.

   Y puede presentar aditamentos como: escalamiento sigma, jerarquías y selección de Boltzman.

   (b) Selección mediante Torneo: Propuesta por *Wetzel* [66]. Se selecciona con base en comparaciones directas entre los individuos. Se subdivide en :

      i. Torneo determinístico.
      ii. Torneo probabilístico.

   (c) Selección de estado uniforme. Propuesta por *Whitley* [67], Reemplaza a los individuos menos aptos de la generación por los más aptos de la nueva generación. Esta técnica se usa en los AGs no generacionales.

5. *Operadores genéticos* (cruza,mutación y elitismo) que alteren la composición de los hijos que se producirán para las siguientes generaciones.

   El AG enfatiza la importancia de la *cruza* sexual (operador principal) sobre el de la *mutación* (operador secundario) y necesita *elitismo* para poder converger al óptimo [55].

   Existen tres tipos principales de cruza:

   - *C*ruza de un punto: se selecciona un punto de manera aleatoria dentro del cromosoma de cada padre y a partir de éste se intercambian los materiales genéticos para dar origen a nuevos individuos [29].
   - *C*ruza de dos puntos: Igual a la anterior excepto que se generan dos puntos de cruza por cada padre [33].
   - *C*ruza uniforme: cruza de n puntos [64, 16].

6. *Valores* para los diferentes *parámetros* que utiliza el algoritmo genético (tamaño de la población, probabilidad de cruza, probabilidad de mutación, número máximo de generaciones, etc.).

El algoritmo general de un algoritmo genético simple [10] está esbozado en la figura 1.3.

begin
  generar `aleatoriamente la población inicial`, $G(0)$
  sea $t := 0$
  repeat
    sea $t := t + 1$
    calcular `la aptitud de cada individuo de` $G(t)$
    seleccionar $G_1(t)$ `en base a la aptitud de` $G(t)$
    aplicar `los operadores genéticos a` $G_1(t)$ `para generar` $G(t+1)$
  until `que cierta condición se satisfaga`
end

Figura 1.3: Algoritmo general de un algoritmo genético simple.

1.6.1 Micro algoritmo genético

El término micro algoritmo genético (micro-AG) se refiere a un algoritmo genético con una población muy pequeña y un proceso de reinicialización. La idea fue sugerida por algunos resultados teóricos obtenidos por Goldberg [27], de acuerdo a las cuales una población de tres individuos era suficiente para converger, sin importar la longitud del cromosoma. El proceso sugerido por Goldberg fue el de aplicar operadores genéticos a una población pequeña (generada aleatoriamente), hasta alcanzar convergencia nominal (es decir, cuando todos los individuos tienen sus genotipos idénticos o muy similares). Posteriormente, debía generarse una nueva población por la transferencia de los mejores individuos de la población obtenida después de dicha convergencia a la nueva. Los restantes individuos serían generados aleatoriamente. La figura 1.4 muestra la representación general de un micro-AG.

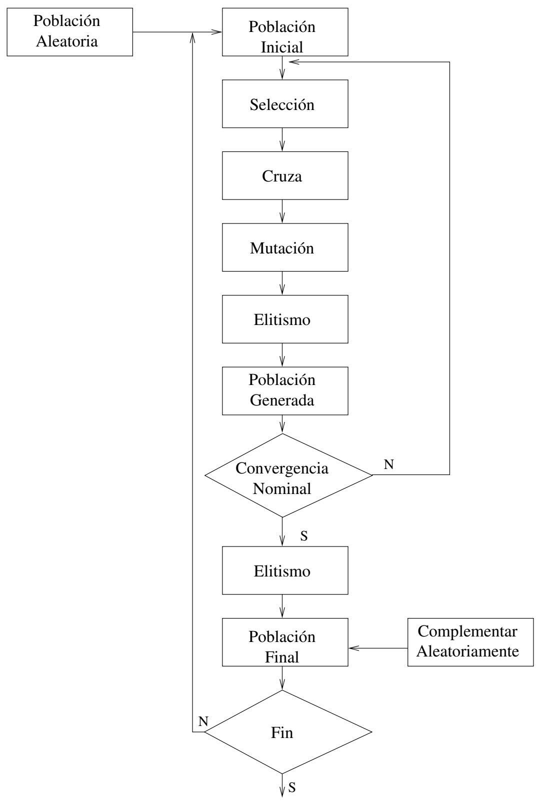


Figura 1.4: Algoritmo general de un micro algoritmo genético.

El primer reporte de una implementación de un micro-AG fue realizado por Krishnakumar [41], quien usó una población de tamaño 5, un porcentaje de cruza de 1 y un porcentaje de mutación de cero. Su aplicación también adoptó una estrategia de elitismo, la cual copiaba la mejor cadena encontrada en la actual población a la siguiente generación. La selección fue desempeñada por el sostenimiento de 4 competidores entre cadenas que eran adyacentes en el arreglo poblacional, y declarando a el individuo con la más alta aptitud como el ganador. Krishnakumar [41] comparó su micro-AG contra un AG simple (con un tamaño de población de 50, un porcentaje de cruza de 0.6 y un porcentaje de mutación de 0.001), y reportó mejores y más rápidos resultados con su micro-AG sobre 2 funciones estacionarias, y un problema de control ingenieril del mundo real. Después de él, otros investigadores han desarrollado aplicaciones con micro algoritmos genéticos [34, 20, 32, 71], aunque su uso ha sido bastante limitado en la literatura especializada.

1.7 Programación genética

Nichal Lynn Cramer [14] y posteriormente, *John R. Koza* [39] propusieron (de forma independiente) el uso de una representación de árbol en la que se implementó un operador de cruza para intercambiar sub-árboles entre los diferentes programas de una población generada al azar (con ciertas restricciones impuestas por la sintaxis del lenguaje de programación utilizado). La diferencia fundamental entre el trabajo de Cramer y el de Koza es que el primero usó una función de aptitud interactiva (es decir, el usuario debía asignar a mano el valor de aptitud de cada árbol de la población), mientras el segundo logró automatizarla, por lo anterior expuesto, la propuesta de Koza llegó a ser la que se adaptara de manera extendida, quedando asentada con el nombre de *Programación genética* (PG) [40]. Hoy en día es muy popular y cuenta con una amplia gama de aplicaciones.

El algoritmo general de la PG [4] se muestra en la figura 1.5:

begin
  generar `aleatoriamente la población inicial,` $G(0)$
  sea $t := 0$
  repeat
    evaluar `los programas en` $G(t)$ `la población existente y asignar un valor de aptitud a cada individuo`
    repeat
      seleccionar G_{temp} `en base a algún proceso de selecci` $G(t)$
      aplicar `los operadores genéticos a` G_{temp}
      insertar `insertar` G_{temp} `en` $G_1(t)$
    until `la nueva población sea completada`
    sea $G(t+1) = G_1(t)$
    sea $t := t+1$
  until `que cierta condición se satisfaga`
  presentar `el mejor individuo de la población`
end

Figura 1.5: Algoritmo básico de la programación genética.

El hecho de que muchos problemas prácticos de diferentes dominios de aplicación puedan ser formulados como un problema de determinación de un *programa solución* que produzca una salida deseada cuando se tienen presentes ciertas entradas particulares, hace a la PG una novedosa línea de investigación. Dentro de algunas de las aplicaciones prácticas se tienen el modelado e identificación de sistemas, procesamiento de señales e imágenes, diseño de circuitos electrónicos, control y robótica, entre otras.

Capítulo 2

Optimización multiobjetivo

2.1 Introducción

La toma de decisiones es algo que no podemos eludir, ocurre diariamente y a cada instante. En la vida cotidiana estos problemas son resueltos mediante el uso del sentido común o intuición, pero en ingeniería se deben resolver eficientemente y en un tiempo razonable. La toma de decisiones puede verse como un problema de optimización en el cual generalmente están involucrados dos o más objetivos (y en la mayoría de las ocasiones éstos están expresados en unidades diferentes y se encuentran en conflicto entre sí). Este problema es conocido como optimización *multiobjetivo* o *multicriterio* (en este trabajo se usarán ambos términos de manera indistinta). Tiene la particular característica de que no existe una solución única, sino un conjunto de soluciones válidas[1] (cada una de ellas puede ser considerada como una solución del problema). Esta característica es llevada a cabo gracias a que ninguna de ellas *domina* a las demás soluciones en todas las funciones objetivo (de ahí que se les llame no dominadas).

En este capítulo se introducen los conceptos básicos de la optimización multiobjetivo, se enuncian algunas técnicas convencionales y finalmente se revisa brevemente el estado del arte en que se encuentra la computación evolutiva en lo que refiere a este tipo de problemas.

2.2 Conceptos básicos y terminología

El problema de optimización multiobjetivo (MOP[2]) es definido por *Osyczka* como [50]:

> El problema de encontrar un vector de **variables de decisión** que satisfaga las **restricciones** y optimice una función vectorial cuyos elementos representen las **funciones objetivo**. Estas funciones forman

[1] A este conjunto de soluciones se les conoce como conjunto óptimo de Pareto.

[2] Por sus siglas en inglés.

una descripción matemática de criterios de desempeño que están usualmente en conflicto entre sí. Por lo tanto, el término optimizar significa encontrar aquella solución que daría un valor aceptable al diseñador en todas las funciones objetivo.

La definición verbal anterior será formalizada posteriormente, pero antes es necesario introducir ciertos conceptos:

Concepto 1 (variables de decisión): Las variables de decisión son un conjunto de n parámetros cuyos valores dan una solución (puede o no ser válida) a un problema de optimización. Estos parámetros serán denotados como $x_j,\ j = 1,2,\ldots,n$. Estas variables de decisión serán representadas en este trabajo de la siguiente manera:

$$\bar{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{2.1}$$

Dicho vector se manejará por conveniencia como:

$$\bar{x} = [x_1, x_2, \ldots, x_n]^T \tag{2.2}$$

Concepto 2 (restricciones): Las restricciones existen generalmente en todos los problemas de ingeniería; ellas delimitan el problema y validan las soluciones. Por lo tanto, se puede decir que las restricciones dibujan el contorno de la región donde se encuentra el conjunto factible del problema.

Las restricciones son funciones de las variables de decisión y pueden ser tanto de igualdad (ecuación (2.4)) como de desigualdad (ecuación (2.3)).

$$r_i \geq 0,\ i = 1,\ldots,m \tag{2.3}$$

$$h_i\ =\ 0,\ i = 1,\ldots,p \tag{2.4}$$

Concepto 3 (función objetivo): Las funciones objetivo forman el criterio de evaluación para saber qué tan buena es una solución; al igual que las restricciones, son funciones de las variables de decisión. En la optimización multiobjetivo existen dos o más funciones objetivos ($f_1(\bar{x}),\ f_2(\bar{x}),\ldots,\ f_k(\bar{x})$) en cada problema. En esta tesis se representarán vectorialmente por $\bar{f}$ de la siguiente manera:

$$\bar{f}(\bar{x}) = \begin{bmatrix} f_1(\bar{x}) \\ f_2(\bar{x}) \\ \vdots \\ f_k(\bar{x}) \end{bmatrix} \tag{2.5}$$

Dicho vector se manejará por conveniencia como:

$$\bar{f}(\bar{x}) = [f_1(\bar{x}), f_2(\bar{x}), \ldots, f_n(\bar{x})]^T \tag{2.6}$$

A continuación se darán tanto una definición matemática para el problema de la optimización multiobjetivo así como algunas definiciones necesarias para comprender de una mejor manera esta problemática:

Definición 1 (el problema de optimización multiobjetivo): Un MOP incluye, un conjunto de n variables de decisión, k funciones objetivo, m restricciones de desigualdad y p de igualdad. En estos problemas el objetivo de la optimización es encontrar el vector de decisión $\bar{x}$ tal que

$$\textit{minimice} \;\; f(\bar{x}) = [f_1(\bar{x}), f_2(\bar{x}), f_3(\bar{x}), \ldots, f_k(\bar{x})]^T \tag{2.7}$$

y cumpla con:

$$\bar{r}_i(\bar{x}) \geq 0 \; i = 1, 2 \ldots, m \tag{2.8}$$

$$\bar{h}_i(\bar{x}) = 0 \; i = 1, 2 \ldots, p \tag{2.9}$$

En palabras esto significa encontrar los parámetros necesarios que optimice la función vectorial objetivo y que satisfagan las restricciones de desigualdad e igualdad.

Definición 2 (conjunto factible) El conjunto factible S está definida como el conjunto de vectores de decisión $\bar{x}$ que satisfacen las restricciones de desigualdad $r(\bar{x})$ y las restricciones de igualdad $h(\bar{x})$:

$$S = \{\bar{x} \in \bar{X} \mid r(\bar{x}) \leq 0 \, \wedge \, h(\bar{x}) = 0\} \tag{2.10}$$

La imagen de S en el espacio de las funciones objetivo está definido por $Z = f(S)$

Definición 3 (dominancia de Pareto): Para dos vectores de decisión $\bar{x}^*, \bar{y}^* \in X$,

$\bar{x}^* \prec \bar{y}^*$	$ssi \; f_i(\bar{x}^*) < f_i(\bar{y}^*)$
$\bar{x}^*$ domina a $\bar{y}^*$	para toda $i = 1, ..., k$
$\bar{x}^* \preceq \bar{y}^*$	$ssi \; f_i(\bar{x}^*) \leq f_i(\bar{y}^*)$
($\bar{x}^*$ domina débilmente a $\bar{y}^*$)	para toda $i = 1, ..., k$
$\bar{x}^* \sim \bar{y}^*$	$ssi \; f_i(\bar{x}^*) \nleq f_i(\bar{y}^*) \, \wedge \, f_i(\bar{y}^*) \nleq f_i(\bar{x}^*)$
($\bar{x}^*$ es indiferente [3]a $\bar{y}^*$)	para toda $i = 1, ..., k$

[3]Son vectores no dominados entre ellos mismos.

Las definiciones son análogas para problemas de maximización ($\succ, \succeq, \sim$). En esta tesis dominancia se referirá a dominancia fuerte.

Es importante notar que aunque la dominancia se da en el espacio de las variables de decisión, la comparación se da en el resultado de la evaluación de las funciones objetivo (espacio de las funciones objetivo). En la figura 2.1 se puede observar la diferencia entre el espacio de las variables de decisión y el espacio de las funciones objetivo.

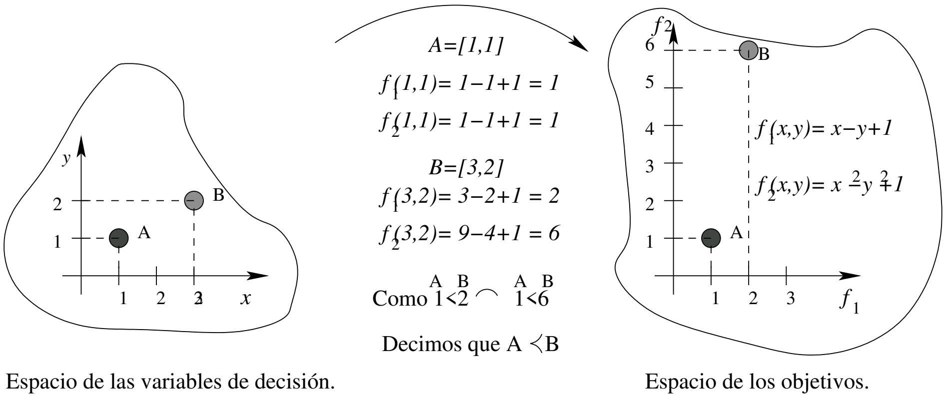


Figura 2.1: Ejemplo que ilustra la diferencia que existente entre el espacio de las variables de decisión (izquierda) y el espacio de las funciones objetivo (derecha).

Tomando un ejemplo de minimización, en el que existen los vectores $\bar{x}^* = [2.4, 5.3, 4.5]^T$, $\bar{y}^* = [2.4, 5.3, 4.8]^T$ y $\bar{z}^* = [3.4, 5.4, 4.7]^T$

entonces de acuerdo a lo anterior expuesto y siendo $f_i, i = 1,2,3$ la función identidad obtenemos que:

$\bar{x}^* \prec \bar{z}^*$ debido a que $2.4 < 3.4,\ 5.3 < 5.4$ y $4.5 < 4.7$,
$\bar{x}^* \preceq \bar{y}^*$ debido a que $2.4 \leq 2.4,\ 5.3 \leq 5.3$ y $4.5 < 4.8$ y
$\bar{y}^* \sim \bar{z}^*$ debido a que $2.4 \leq 3.4,\ 5.3 \leq 5.4$ y $4.8 > 4.7$.

Definición 4 (optimalidad de Pareto): Un vector de decisión $\bar{x}^*$ es óptimo de Pareto si y sólo si:

$$\bar{x}^* \in S \mid \neg\exists\, \bar{y}^* \in S \mid \bar{f}(\bar{y}^*) \preceq \bar{f}(\bar{x}^*) \tag{2.11}$$

En palabras podemos decir que un vector de decisión que es miembro del conjunto factible S es óptimo de Pareto si no existe otro vector de decisión $\bar{y}^*$ que pertenezca a S y que lo domine.

Definición 5 (conjunto de Óptimos de Pareto): El conjunto de óptimos de Pareto (P^*) esta definido como:

$$P^* = \{\bar{x}^* \in S \mid \neg\exists\, \bar{y}^* \in S \mid \bar{f}(\bar{y}*) \preceq \bar{f}(\bar{x}^*)\} \tag{2.12}$$

El óptimo de Pareto se encuentra en el espacio de las variables de decisión, y se está representado gráficamente tanto en la figura 2.2 como en la figura 2.4.

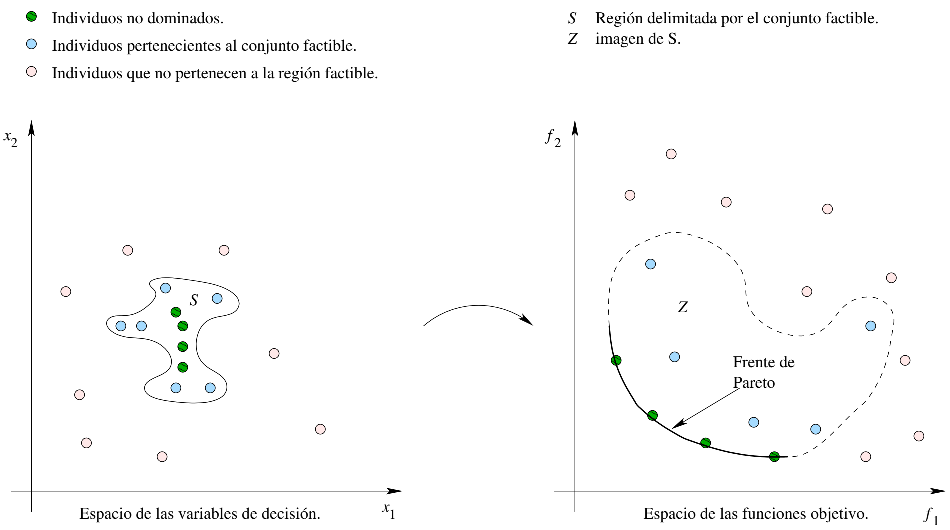


Figura 2.2: Ejemplo ilustrativo del conjunto factible S y del conjunto óptimo de Pareto en el espacio de las variables de decisión (izquierda), y de sus imágenes en el espacio de las funciones objetivo (derecha).

Definición 6 (frente de Pareto) : Para un MOP $\bar{f}(x)$ dado y un conjunto óptimo de Pareto P^*, el frente de Pareto (PF) se define como:

$$PF^* = \{\bar{u} = \bar{f} = (f_1(x), \ldots, f_k(x)) | x \in P^*\} \tag{2.13}$$

El óptimo de Pareto se encuentra en el espacio de las variables de decisión, y se está representado gráficamente tanto en la figura 2.2 como en la figura 2.4.

Cabe mencionar que algunos problemas presentan diferentes frentes de Pareto ficticios que atraen la mayoría de las soluciones, estos frentes son conocidos como frentes de Pareto locales. En la figura 2.3 se ilustra gráficamente la diferencia entre un frente de Pareto local y uno global.

A continuación se brinda el problema de Belegundu [5] que tiene las características básicas de un típico problema de optimización multiobjetivo, y se muestran las gráficas que corresponden tanto al espacio de las variables de decisión, como de las funciones objetivo.

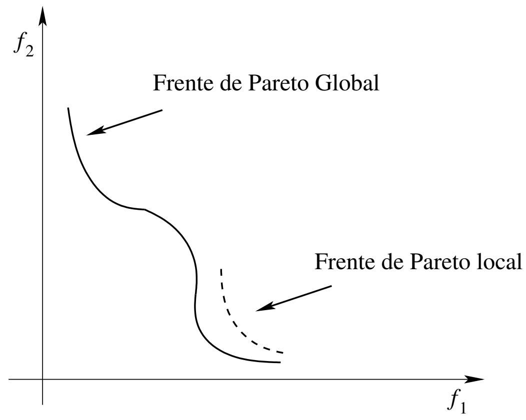


Figura 2.3: Gráfica que ilustra la diferencia entre el frente de Pareto local y global

$$F = (f_1(x,y), f_2(x,y)) \tag{2.14}$$

donde:

$$\begin{aligned} f_1(x,y) &= -2x + y \\ f_2(x,y) &= 2x + y \end{aligned} \tag{2.15}$$

sujeto a:

$$\begin{gathered} 0 \leq x \leq 5, 0 \leq y \leq 3 \\ -x + y - 1 \leq 0, \\ x + y - 7 \leq 0 \end{gathered} \tag{2.16}$$

2.3 Técnicas para optimización multiobjetivo

Desde que el economista *Vilfredo Pareto* introdujera en 1896 el concepto de *solución compromiso* se han desarrollado un gran número de técnicas de optimización multiobjetivo tanto tradicionales (investigación de operaciones), como alternativas (algoritmos evolutivos, recocido simulado y búsqueda tabú entre otros).

A continuación se hace una breve revisión de los principales métodos tradicionales y evolutivos multiobjetivo.

2.3.1 Técnicas tradicionales

La comunidad de investigación de operaciones ha desarrollado más de veinte técnicas de optimización multiobjetivo [51, 47], de entre las cuales discutiremos a continuación las principales.

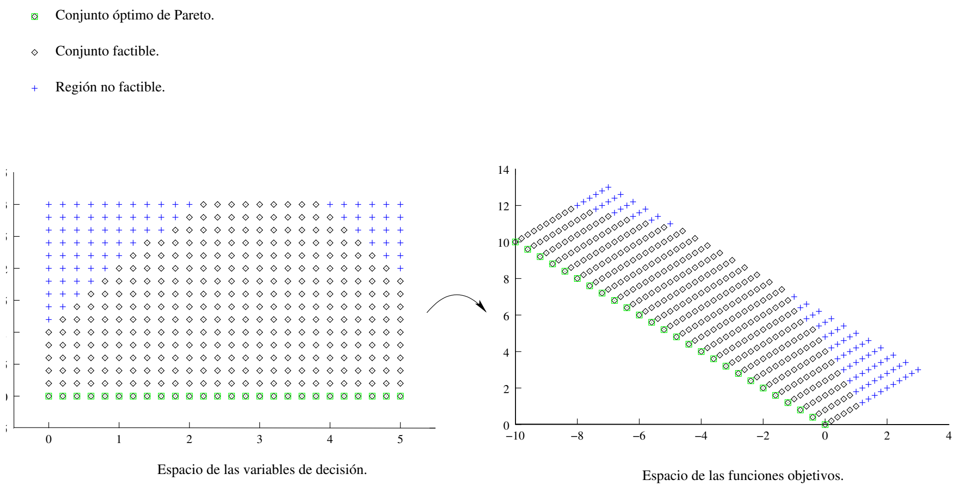


Figura 2.4: Gráficas del espacio de las variables de decisión (izquierda), y del espacio de las funciones objetivo (derecha) de la función de Belegundu [5].

2.3.1.1 Métodos sin preferencias

En los métodos sin preferencias se incluyen todos aquellos métodos que no toman en consideración las opiniones de preferencias de quien toma las decisiones. El problema de optimización multiobjetivo es resuelto usando algunos métodos relativamente simples y la solución encontrada es presentada al encargado de tomar las decisiones, el cual puede aceptar o rechazar la solución. Estos métodos son generalmente utilizados cuando el encargado de la toma de decisiones no espera una solución especial y está satisfecho con cualquier solución óptima en el sentido de Pareto.

Ejemplos de esta clasificación son:

- Método del criterio global [52, 44].
- Método multiobjetivo de los paquetes próximos [48].

2.3.1.2 Métodos a posteriori

Estos métodos generan un conjunto de óptimos de Pareto, con el inconveniente de que el proceso de generación es usualmente muy costoso computacionalmente y en la mayoría de los problemas es sumamente difícil alcanzarlo.

Estos métodos se dividen en dos subclases: los que pueden encontrar todo el frente de Pareto, y lo que pueden generar únicamente los puntos que se encuentran en los extremos del frente de Pareto.

Ejemplos de métodos a posteriori:

- Método de suma de pesos [56, 43].
- Método de restricciones ε [72].

2.3.1.3 Métodos a priori

La especificación de preferencias en estos métodos se realiza antes del proceso de solución. La principal dificultad de estos métodos radica en que no siempre se saben de antemano las características deseables que deben de tener las soluciones del problema. A continuación se muestran sus tres principales exponentes:

- Método de la función valor [35].
- Método Lexicográfico [21].
- Programación de metas [1]

2.3.1.4 Métodos Interactivos

Esta clase es la más desarrollada de las cuatro. Esto se debe a que se solventan la mayoría de los problemas de la toma de decisiones mediante la interacción entre quién toma la decisión y el método interactivo. El algoritmo básico de los algoritmos [47] se puede encontrar en la figura 2.5.

begin
  encontrar `una solución factible`
  repeat
    interactuar `con quien toma las decisiones` (*introducción de preferencias*)
    obtener `una nueva solución de acuerdo a esas preferencias`
  until `se acepte una solución`
end

Figura 2.5: Algoritmo básico de los algoritmos interactivos

A continuación se nombran algunos de los métodos pertenecientes a esta clase:

- Método Interactivo de compromisos valuados [11].
- Método de satisfacción de compromisos [49].
- Método de Geoffrion-Dyer-Feinberg [3].
- Técnica secuencial de optimización próxima [57].
- Método de Tchebycheff [63].
- Método de punto de referencia [68].
- Método GUESS [9].
- Búsqueda Light Beam [2].
- Enfoque de referencia de dirección [38].
- Método NIMBUS [46].

2.3.1.5 Desventajas de las técnicas convencionales

Los métodos tradicionales son bastante limitados y generalmente son lo suficientemente costosos como para obtener una respuesta en un tiempo polinomial a medida que el problema crece. En caso de que se encuentren las soluciones no dominadas, éstas suelen corresponder a una única solución o a una porción limitada del frente de Pareto esperado.

2.3.2 Técnicas evolutivas

Desde que *Rosenberg* [54] hiciese alusión al potencial de los algoritmos evolutivos para resolver problemas multiobjetivo el interés de esta área por parte de la comunidad de computación evolutiva ha crecido enormemente, hasta llegar a afianzarse como una línea de investigación propia, llamada *optimización evolutiva multiobjetivo*[4].

Este interés está justificado debido a que la gran mayoría de los problemas del mundo real son realmente problemas de optimización multiobjetivo, si bien suelen convertirse a problemas con una sola función objetivo (volviendo restricciones las demás funciones objetivo). Además, los algoritmos evolutivos tienen la capacidad inherente de encontrar el conjunto Pareto en un tiempo razonable y en una sola ejecución, en contraste con las técnicas tradicionales y la mayoría de las otras heurísticas para optimización multiobjetivo, que suelen generar una sola solución a la vez.

Existe un número considerable de técnicas evolutivas para optimización multiobjetivo (aunque existe una comunidad robusta, solo existe un puñado de revisiones del estado del arte [25, 65, 12]). *Coello* [12] las agrupa en:

- Formas simplistas:
  - Suma de pesos.
  - Programación de metas.
  - Satisfacción de metas.
  - Método de restricciones ε.
- Técnicas no basadas en óptimos de Pareto.
  - VEGA.
  - Ordenamiento Lexicográfico.
  - Uso de géneros para identificar objetivos.
  - Uso de Min/Max con pesos.
  - Algoritmo genético no generacional.

[4] El repositorio de EMOO en http://www.lania.mx/c̃coello/emoo

  - Uso de pesos generados aleatoriamente y elitismo.

- Técnicas basadas en óptimos de Pareto.
  - MOGA.
  - NSGA.
  - NPGA.
  - NSGA II.

A continuación se hace una breve descripción de las características más importantes de las técnicas que representan de mejor manera el estado actual de la computación evolutiva.

2.3.2.1 Vector Evaluated Genetic Algorithm, VEGA

GENESIS [28] sirvió de base para que *David Schaffer* [59] modificara la manera en que la selección era hecha y generara un nuevo método con la capacidad de manejar objetivos múltiples, el cual ahora es conocido como VEGA.

La selección tras la modificación al genético simple quedó como sigue:

> Se genera un número de sub-poblaciones igual al número de objetivos (k), y se efectúa selección proporcional de acuerdo al objetivo correspondiente. Posteriormente se mezclan las sub-poblaciones obtenidas para generar una nueva única población a la cual se le aplica los operadores genéticos convencionales (cruza y mutación).

2.3.2.2 Multiobjective Genetic Algorithm, MOGA

Ideada por *Fonseca y Fleming* [24]. En esta técnica la jerarquía de un individuo está dada por el número de individuos que lo dominan en la población. Tomando esto en cuenta, los individuos no dominados tendrán asignada la jerarquía 1, y se desciende proporcionalmente conforme éstos sean dominados por más individuos. La posición en la jerarquía está dada por [24]:

$$Jerarquía(x_i, t) = 1 + p_i^t \tag{2.17}$$

donde:

x_i = individuo correspondiente al subíndice i.

t = la generación actual.

p_i^t = número de individuos que dominan al individuo perteneciente al subíndice i en la generación t.

2.3.2.3 Non-dominated Sorting Genetic Algorithm, NSGA

Fue propuesta por *Srinivas y Deb* [61] y se basa en la clasificación de la población mediante el uso de varias capas jerárquicas. Los individuos son jerarquizados de acuerdo a su no dominancia, de tal manera que los individuos no dominados con respecto a toda la población pertenecen a la primera capa. Para saber qué individuos pertenecen a la segunda capa se eliminan los individuos no dominados de la primer capa (para efectos de esta comparación) y los (ahora) individuos no dominados pertenecen a la segunda capa. El proceso se repite iterativamente hasta colocar a toda los individuos en sus capas correspondientes. Al concluir esto se realiza selección proporcional, de tal manera que los individuos pertenecientes a la primer capa tienen una mejor posibilidad para ser seleccionados una mayor cantidad de veces. En la figura 2.6 se muestra el uso de las capas.

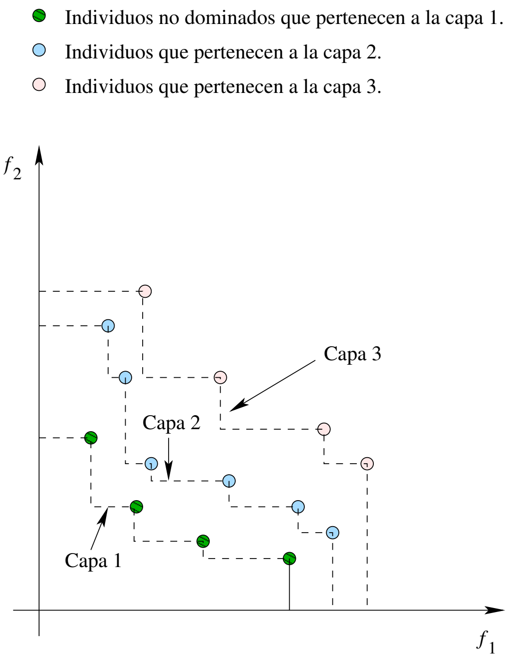


Figura 2.6: Representación gráfica del manejo de las capas por del NSGA.

Existe una segunda versión de este algoritmo, denominado *NSGA II* propuesto recientemente por *Deb et al* [18], el cual solventa muchos de los problemas de la versión original del algoritmo [12].

2.3.2.4 Niched Pareto Genetic Algorithm, NPGA

Fue propuesta por *Horn y Nafpliotis* [30]. Esta técnica utiliza selección mediante torneo basada en dominancia de Pareto en un subconjunto de la población. La metodología consiste en comparar a dos individuos contra un subconjunto de la población (la cantidad de elementos de dicho subconjunto está dado por el parámetro tamaño de torneo), cuando estas comparaciones no arrojan a un vencedor (ambos individuos quedaron empatados) se utiliza *repartición de aptitud* [26, 30]. La elección de un tamaño de torneo inadecuado hace que disminuya el desempeño del algoritmo.

2.3.2.5 Pareto Archived Evolution Strategy, PAES

Fue propuesta por *Knowles* y *Corne* [37]. PAES es una estrategia evolutiva en su forma más simple: la *(1+1)-EE*. Utiliza búsqueda local y un archivo histórico para almacenar las soluciones no dominadas que va encontrando a lo largo de la corrida. Para mantener la diversidad apropiada en este archivo histórico se introdujo la idea de una malla adaptativa. Esta malla tiene visibles ventajas sobre el uso de nichos debido a que computacionalmente tiene un menor costo, y debido a su adaptabilidad no necesita parámetros extra que son claves y necesarios para el uso de nichos y además afectan significativamente al desempeño del algoritmo.

2.3.2.6 Tendencia actual

Un problema de algunas de las técnicas anteriormente descritas radica en el uso de un proceso de jerarquización para clasificar a los individuos de una población basado en dominancia, y es precisamente este procedimiento el que consume más tiempo en una corrida (es un proceso $O(kM^2)$, donde k es el número de funciones objetivo y M es el tamaño de la población). Adicionalmente a esto, un mecanismo extra es requerido para preservar la diversidad en la población (suelen adoptarse algunas formas de compartición de aptitud, lo cual generalmente implica el uso de procesos $O(M^2)$). Sin embargo, recientes investigaciones han mostrados caminos para mejorar la eficiencia de una técnica para optimización multiobjetivo. Algunos investigadores se han enfocado en (a) reducir el chequeo de no dominancia y en (b) el desarrollo de algoritmos eficientes para mantener diversidad. En cuanto al primer punto, el principal énfasis ha sido sobre el uso de un archivo externo que guarda los vectores no dominados encontrados durante el proceso evolutivo los cuales son reinsertados después en la población (esto puede ser visto como una forma de elitismo en el contexto de optimización multiobjetivo). Retomando el segundo punto el principal énfasis ha sido en el uso de técnicas de agrupamiento o algoritmos basados en posicionamiento geográfico de individuos en una malla adaptativa.

Algunos investigadores han sugerido también el uso de un algoritmo genético distribuido en el cual la dominancia se aplica solamente a vecinos dentro de cierta región. Este tipo de algoritmo pueden solventar los dos problemas previamente mencionados en forma simultánea. En esta técnica la dominancia en el frente de Pareto se debe de aplicar en paralelo a pequeños grupos de individuos y la diversidad no requiere un mecanismo extra, ya que su naturaleza emerge de la población distribuida. Sin embargo, para tomar ventajas de estas características es necesaria una arquitectura paralela.

2.3.2.7 Ventajas de los algoritmos evolutivos

Los algoritmos evolutivos son especialmente apropiados para resolver problemas de optimización multiobjetivo debido a su gran flexibilidad, adaptabilidad y sólido desempeño. Estas características hacen posible que sea capaz de lidiar con diferentes formas del frente de Pareto (desconectados, convexos o cóncavos), sin que se degrade su desempeño. También le permite localizar y poblar zonas promisorias en problemas con grandes espacios de búsqueda mientras que las técnicas tradicionales no podrían acercarse siquiera a la zona factible de muchos de ellos (generalmente asumen que el frente de Pareto es convexo y que las funciones objetivo son diferenciables). Adicionalmente a esto los algoritmos evolutivos tienen la capacidad inherente de encontrar diferentes miembros del conjunto de óptimos de Pareto en una sola corrida, gracias en gran medida al hecho de que son una técnica poblacional y no una técnica local como el recocido simulado o la búsqueda tabú.

Capítulo 3

Descripción de la técnica

3.1 Introducción

El algoritmo aquí presentado fue desarrollado tomando la tendencia actual que dicta el desarrollo de técnicas eficientes mediante la reducción del chequeo de no dominancia(1) y el posicionamiento geográfico(2) como mecanismo para mantener diversidad en el frente de Pareto. Aunado a esta tendencia, este algoritmo está cimentado sobre la convicción de que mediante varias formas de elitismo(3) se puede obtener un frente de Pareto con una mejor calidad de soluciones y menor costo computacional.

Los tres puntos anteriormente citados que forman el cimiento ideológico para la concepción de este algoritmo maduraron en lo siguiente:

1. Para reducir el chequeo de no dominancia entre toda la población se ideó una manera de tener pocos individuos para ser comparados. Esta búsqueda dio origen a la implementación de un micro algoritmo genético (micro-AG) para lidiar con estos problemas, admás de incluir un archivo histórico.

2. Se adoptó una malla adaptativa para ser usada como mecanismo de posicionamiento geográfico para mantener la diversidad en las soluciones no dominadas encontradas de una manera sencilla y eficiente.

3. La convicción elitista dio origen a la introducción de tres tipos de elitismo en este trabajo.

En el presente capítulo se presenta detalladamente el algoritmo propuesto. Inicialmente, se introduce el algoritmo general; posteriormente, se detallan cada una de sus componentes y se finaliza describiendo los parámetros utilizados por el algoritmo.

3.2 Algoritmo general

El sistema propuesto trabaja a grandes rasgos de la siguiente manera (puede observarse gráficamente en la figura 3.1):

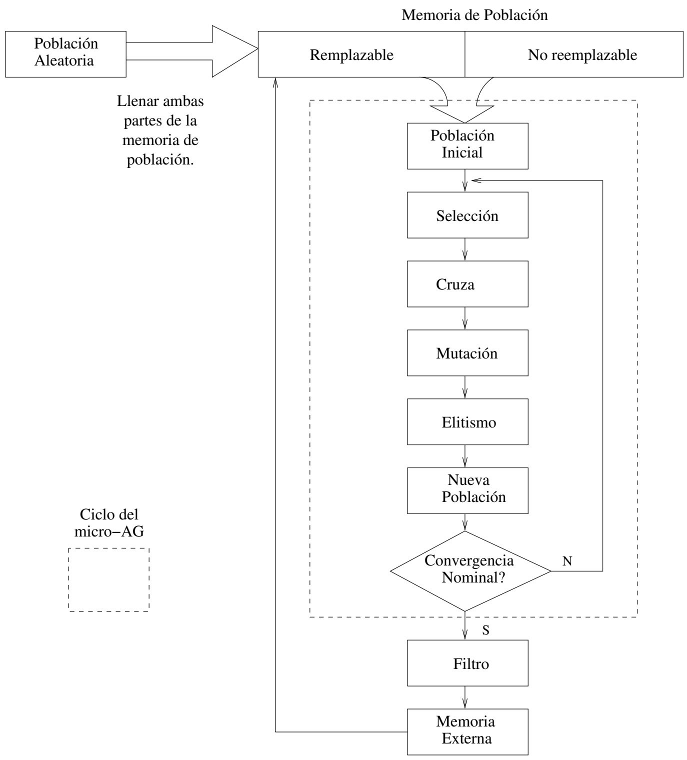


Figura 3.1: Sistema propuesto.

Inicialmente, se llena la memoria de población$^{(a)}$ P con individuos generados aleatoriamente. Esta memoria está dividida en dos partes, una reemplazable P^r y otra no reemplazable P^{nr}. De esta memoria el micro-AG$^{(b)}$ irá obteniendo una pequeña población (de trabajo) P^i para irla evolucionando. Tras la ejecución del micro-AG$^{(b)}$ se realiza un $filtro^{(c)}$, a manera de eliminar las soluciones *dominadas* y finalmente quedarse con sólo dos soluciones S_1, S_2[1] *no dominadas*. Posteriormente estas se comparan contra un par de contendientes de P^r. Si S_1 o S_2 dominan a sus adversarios, ocupan su lugar en P^r y finalmente tanto S_1 como S_2 se insertan en el archivo *histórico*$^{(d)}$[2](si es que así procede). Comienza el ciclo otra vez, hasta alcanzar la condición necesaria de paro.

La evolución de P^i por parte del micro-AG es realizada en forma tradicional: Se efectúa selección, cruza, mutación y elitismo hasta que se alcanza convergencia nominal (en nuestro caso la convergencia nominal se alcanza después de cierto

[1]Si es que existe más de una solución no dominada.

[2]También conocido como archivo de soluciones no dominadas o archivo externo.

número de iteraciones).

A continuación se describirán a detalle los puntos estratégicos mencionados anteriormente:

a. Memoria de población.

b. Micro-AG.

c. Filtro.

d. Malla adaptativa.

3.3 Memoria de población

Debido al ruido estocástico, un algoritmo genético tenderá a converger, con el tiempo, a una solución única. Para que un algoritmo genético usado para optimización multiobjetivo no pierda diversidad, es necesario que la población del mismo no converja hacia una porción única del frente de Pareto o hacia un frente de Pareto local. Para lograr esto se ideó una memoria de población, la cual es inicializada aleatoriamente y está dividida en dos partes: una porción reemplazable (P^r) y, otra no reemplazable (P^{nr}). Una conceptualización gráfica de esta memoria se muestra en la figura 3.2.

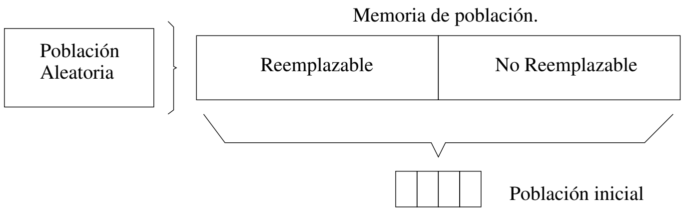


Figura 3.2: Memoria de población.

P^{nr} nunca cambiará durante la ejecución completa del algoritmo, y será la encargada de mantener la diversidad requerida por el algoritmo. En cambio, P^r experimentará cambios después de cada ciclo del micro-AG como se indicará más adelante (en la sección *3.6*).

La población (de trabajo) P^t del micro-AG al principio de cada uno de sus ciclos es tomada (con cierta probabilidad) de ambas porciones de la memoria de población, por lo que al final se tiene una mezcla de individuos: los generados aleatoriamente (P^{nr}), y los individuos evolucionados (P^r). La idea es que la recombinación de ambas clases den un poder exploratorio robusto, ya que P^{nr} no permitirá el sesgo de toda la población hacia una determinada región en el espacio de búsqueda, lo que dará la diversidad deseada a la población.

3.4 Micro-AG, cuerpo

Durante cada ciclo, el micro-AG utiliza operadores genéticos convencionales, mismos que se describen brevemente a continuacion. [3]

- **Selección**
  Se usa selección por torneo binario utilizando dominancia como método de comparación. Esta selección fue propuesta por *Wetzel* [66] y estudiada en la tesis doctoral de *Brindle* [8]. La idea básica del método es seleccionar con base en comparaciones directas de los individuos de la población. Cada competencia requiere la selección aleatoria de un número constante de individuos de la población. La comparación entre estos individuos puede realizarse en tiempo constante requiriéndose n competencias de este tipo para complementar una generación. Por lo tanto, el algoritmo es $O(n)$. La versión utilizada en este caso fue la determinista: Esta versión garantiza que el mejor individuo será seleccionado P veces y tiene el siguiente algoritmo:

  - Barajar los individuos de la población.
  - Escoger un número de P individuos (típicamente 2).
  - Compararlos con base en su aptitud.
  - El ganador del torneo es el individuo más apto.
  - Se debe de barajar la población un total de P veces para seleccionar N padres.

- **Cruza**
  Para el desarrollo del micro-AG se implementó la cruza de 2 puntos [33] (representada en la figura 3.3), debido a que esta cruza es la menos disruptiva de las de n puntos [64, 16], y evita algunos de los problemas de la cruza de un solo punto [29].

- **Mutación**
  Para esta técnica se utilizó mutación uniforme, la cual mantiene un porcentaje de mutación constante a lo largo de todo el proceso evolutivo. La mutación uniforme se ilustra en la figura 3.4.

- **Elitismo**
  Solamente un vector no dominado es arbitrariamente seleccionado de la población en cada generación y copiado intacto a la siguiente [55].

Después de que el cuerpo del micro-AG termina una ejecución (esto se ejemplifica en la figura 3.5), se *filtran* los individuos y comienza otra vez este proceso hasta que una condición final de paro sea alcanzada.

[3] Las descripciones de los operadores genéticos fueron tomadas y conceptualizadas de [13]

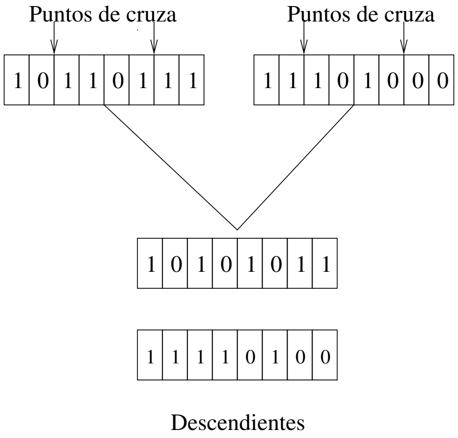


Figura 3.3: Figura que ilustra la cruza de dos puntos.

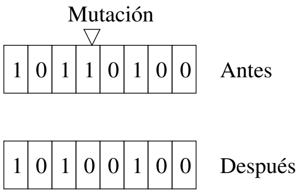


Figura 3.4: Representación gráfica de la mutación uniforme

3.5 El Filtro

El uso de un filtro fue motivado por la condición elitista con que el micro-AG fue concebido. El uso del filtro no únicamente corresponde con el hecho de dejar pasar a los individuos no dominados S_1 y S_2 (si es que existe más de un individuo no dominado) arrojados por el proceso evolutivo hacia un archivo histórico, sino que también conlleva a que si renovamos la memoria de población (reemplazable) por soluciones no dominadas S_1 o S_2, en caso de que éstas dominen a las correspondientes alojadas en la memoria de población, gradualmente, ciclo tras ciclo, la población irá adquiriendo una mejor aptitud, teniendo una mejor oportunidad de converger al frente de Pareto verdadero (o global).

Este filtro también tiene otra importante labor elitista y en extremo necesaria para que esta técnica funcione adecuadamente. Esta labor es denominada *ciclo de reemplazo, o segundo elitismo* y como su nombre lo sugiere, únicamente se realiza a un cierto intervalo preestablecido. El ciclo de reemplazo toma puntos (soluciones no dominadas) correspondientes a todas las regiones del frente de Pareto obtenido hasta el momento. La memoria (reemplazable) es renovada con estos puntos. Esta labor está fundamentada sobre el mismo principio que la inmediata anterior, con la diferencia que en la tarea anterior no se garantiza que la memoria tenga soluciones no dominadas globalmente, mientras que en ésta sí. Es aconsejable de que esta última tarea únicamente se realice cuando el frente de Pareto obtenido tenga al me-

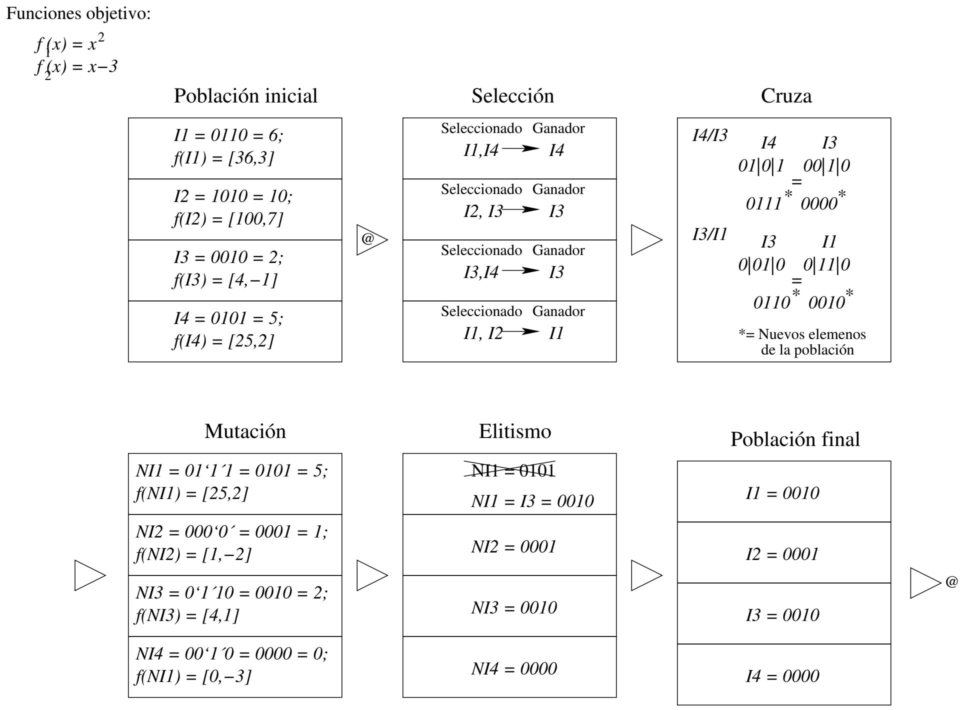


Figura 3.5: Ejemplo de un ciclo del cuerpo del micro-AG

nos los mismos individuos que la memoria reemplazable, ya que en caso contrario se puede tener convergencia prematura.

3.6 La malla adaptativa

La labor de la malla adaptativa es el archivar históricamente a los vectores no dominados y se divide en dos procesos:

1. El controlador de archivado y
2. La malla adaptativa (como un subproceso).

3.6.1 Controlador de archivado

La función del controlador de archivado es la de tomar la decisión si una solución es alojable dentro del archivo externo. En caso de que una solución sea potencialmente alojable pero el límite de sus funciones no le permiten decidir, entonces deja la decisión a la malla adaptativa. La metodología del controlador de archivado es la siguiente:

Después de que el micro-AG (cuerpo) termina un ciclo, se escogen los vectores no dominados de la población final y se comparan uno por uno contra el frente de Pareto actual alojado en la memoria externa. En caso de que no hubiera aún ningún vector en el archivo externo entonces la solución actual es aceptada (ver caso 1 de la figura 3.6). En el caso de que esta nueva solución sea dominada por algún individuo del archivo externo, se le descarta automáticamente (ver caso 2 de la figura 3.6), si no es así y si ninguno de los miembros del archivo externo dominan a dicha solución, entonces se procede a almacenarla. Si ésta domina a vectores pertenecientes al archivo externo, se eliminan los vectores dominados (ver el caso 3 y 4 de la figura 3.6). Finalmente, si el archivo ha alcanzado el máximo de soluciones permitidas, entonces se invoca la malla adaptativa (ver caso 5 de la figura 3.6).

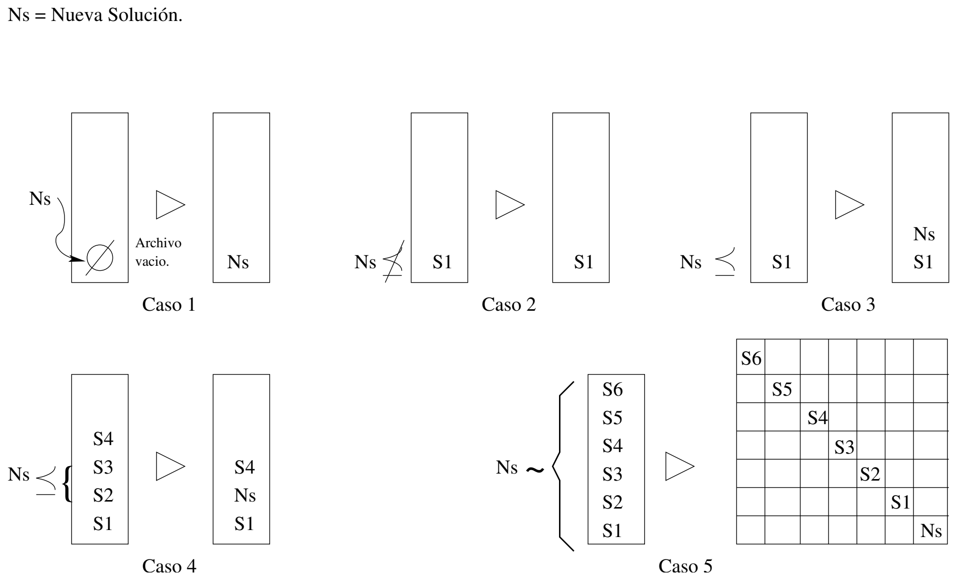


Figura 3.6: Casos que se pueden presentar en el controlador de archivado.

El proceso anterior descrito es detallado en el pseudocódigo mostrado en la figura 3.7.

3.6.2 Malla adaptativa

Para mantener diversidad en el frente de Pareto, se usa un algoritmo similar a la malla adaptativa propuesta por *Knowles y Corne* [37]. La idea básica es que una vez que el archivo donde se guardan las soluciones no dominadas ha llegado a su límite, se dividen las soluciones no dominadas en diferentes regiones[4].

Esta división *solución-región* es realizada con el objetivo de alojar a los individuos no dominados de una manera uniforme y distribuida a lo largo del frente de Pareto. Esto se realiza aceptando únicamente a los individuos que pertenecen a regiones que esten menos pobladas (esto se muestraen la figura 3.8), o en el caso que algún individuo pertenezca a una región fuera de los límites previamente

[4]los términos “región” e “hipercubo” se usarán indistintamente en este trabajo.

```
function Controlador-Archivado(solución)
  begin
    if archivo-externo E = ϕ
    then almacenar solución en E
    else if solución domina a E
    then elimina a dominados
      almacenar solución en E
    else if solución no es dominada por E
    then if E no está lleno
      almacenar solución en E
      else almacenar solución en E
        actualizar localidades
      else Malla − Adaptativa(solución)
  end function
```

Figura 3.7: Pseudocódigo del controlador de archivado

especificados para la malla (ver la figura 3.9).

El costo computacional fue la principal motivación para usar esta malla adaptativa ya que es menor que cualquier proceso de jerarquización establecido previamente y también necesita menos parámetros (el tamaño esperado del frente de Pareto y la cantidad de posiciones en la que se dividirá el espacio de soluciones para cada objetivo). Este ahorro computacional se debe a que el proceso de determinar la posición geográfica de cierto individuo tiene un costo bajo (el mecanismo está descrito en la ecuación 3.4, y su representación gráfica se muestra en la figura 3.8). Sin embargo, cuando los individuos están fuera de rango se tienen que reasignar todas las posiciones (como se ejemplifica en la figura 3.9). Esta reasignación global de posiciones no ocurre tan a menudo gracias a que se aloja cierta cantidad extra de espacio en el primero y en la última posición de la malla para minimizar la ocurrencia de este proceso (ver figura 3.9).

3.6.2.1 Generación de la malla adaptativa

La malla adaptativa es en realidad un espacio formado por hipercubos k-dimensionales, estos hipercubos tienen tantas componentes como funciones k objetivo existan. Cada hipercubo se puede interpretar como una región geográfica que contiene a un número *no* determinado de individuos y la cantidad total de individuos trata de abarcar uniformemente la mayor cantidad de hipercubos posible. Para que esto sea posible es necesario conocer y poder obtener ciertos datos como los límites de la malla, las dimensiones de cada región, la cantidad poblacional por región y finalmente saber a que región pertenece cada solución (incluyendo la nueva solución que se pretende ingresar).

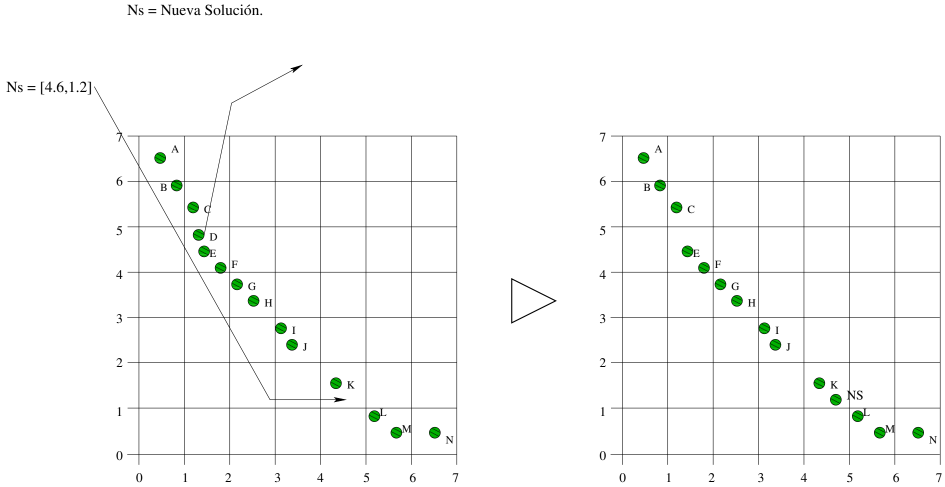


Figura 3.8: Representación gráfica de la inserción de un nuevo elemento en la malla adaptativa.

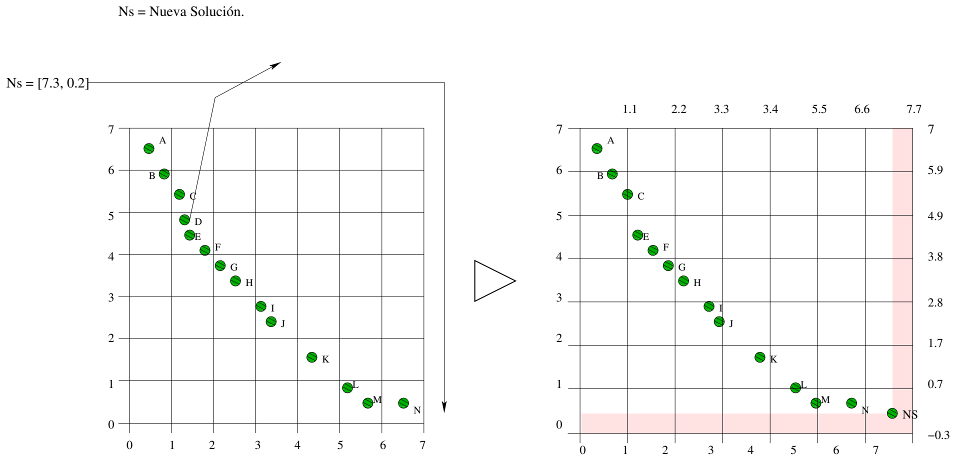


Figura 3.9: Representación gráfica de la inserción de un nuevo elemento en la malla adaptativa y su reajuste para cubrir la nueva solución.

A continuación se describirán la forma en que se obtienen algunos de estos datos:

Límites de la malla: Las dimensiones de la región que la malla forzosamente debe de cubrir esta delimitada por los mejores y peores valores de aptitud que tiene la población para cada objetivo[5]. El límite superior de cada componente de esta región esta dado por la ecuación 3.1 ($mAptObj_i$, menor aptitud para el objetivo i), y el inferior por la ecuación 3.2 ($MAptObj_i$, mayor aptitud para el objetivo i).

$$mAptObj_i = max(f_i(s);\ s \in P^*),\ i = 1,2,\ldots,k \tag{3.1}$$

$$MAptObj_i = min(f_i(s);\ s \in P^*),\ i = 1,2,\ldots,k \tag{3.2}$$

donde:

k = Cantidad total de objetivos.

P^* = Conjuntos de óptimos de Pareto encontrado.

Sin embargo con el propósito de minimizar la cantidad de recálculo de la malla, es necesario que ésta abarque más espacio que el necesario por cubrir. Este espacio extra es asignado por componente de la malla y es igual a las dimensiones de un hipercubo, esto es mostrado en la figura 3.10 (la longitud de cada componente del hipercubo es dividida entre dos para que exista un límite extra en el extremo inferior, y otro en el extremo superior).

Dimensiones del hipercubo: El tamaño de la división (componente) i de cada región ($tamDiv_i$) es obtenido de acuerdo a la siguiente ecuación:

$$tamDiv_i = (MAptObj_i - mAptObj_i)/(nDivs - 1);\ i = 1,2,\ldots,k \tag{3.3}$$

donde: k = Cantidad total de objetivos.
$nDivs$ = Número de divisiones en que se dividirá el espacio de las soluciones.

Identificación de la región La región ($LOCs$) a la que pertenece la solución s es obtenida por:

$$LOCs = \sum_{i=1,j=k}^{k,1} \frac{aptObj_i - inicioMalla_i}{tamDivision_i} * numdivs^j \tag{3.4}$$

Donde:
k = Número de objetivos.
$aptObj_i$ = Aptitud del individuo para el objetivo i.

[5]La malla se realiza en el espacio de los objetivos.

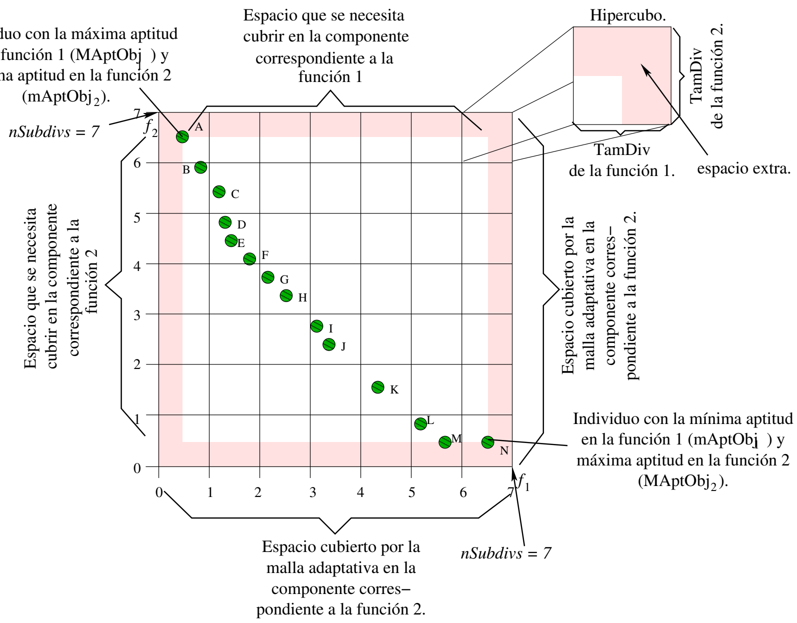


Figura 3.10: Representación gráfica para la malla adaptativa, en este caso se utilizaron dos funciones objetivo para su mejor comprensión.

> $inicioMalla_i$ = Inicio de la malla para el objetivo i.
> $tamDiv_i$ = Tamaño de la división para el objetivo i.
> y $numdivs$= Número de divisiones en que se dividirá el espacio de las soluciones.

El pseudocódigo de la malla adaptativa se despliega en la figura 3.11.

El pseudocódigo general de la técnica anteriormente descrita esta conjuntado en la figura 3.12.

3.7 Manejo de restricciones

El manejo de restricciones por parte del micro-AG es llevado a cabo mediante un pequeño mecanismo implementado en la selección, la malla adaptativa y memoria de población.

En la selección, el manejo de restricciones se lleva a cabo al momento de checar la dominancia de Pareto. Esto es, cuando se comparan dos individuos, primero se comparan las restricciones violadas y el individuo dominante será quién menos restricciones viole. En el caso de que ambos individuos hayan obtenido el mismo número de restricciones violadas se procede a realizar la comparación mediante la dominancia de Pareto tradicional.

Nunca se guardará a un individuo en el archivo histórico si éste ha violado alguna restricción. Sin embargo, es posible que dicho individuo sí pueda ser alojado

function Malla-Adaptativa(`solución`)
  begin
    `solución`$_1 = x; x \in$ `La región más poblada` P
    if `solución esta fuera de rango`
    then `elimina solución`
      `almacenar solución en` E
      `actualiza localidades`
    else if `solución`$_1$ `en region más poblada que solución`
    then `elimina solución`$_1$
      `almacenar solución en` E
      `actualiza localidades a la que pertenecía solucion`$_1$ `y`
       `a la que pertenece solución`
  end function

Figura 3.11: Pseudocódigo de la malla adaptativa

dentro de la memoria de población P^r (si es que domina a su competidor) para tener la posibilidad de evolucionarlo y de que contribuya a llegar a la zona factible.

3.8 Descripción de parámetros

En esta sección se hará una descripción de los parámetros utilizados por el sistema propuesto.

int itot Hace referencia a la condición de paro del micro-AG (número de iteraciones). Variar de acuerdo a la complejidad del algoritmo.

int gmax Este parámetro es el encargado de manejar la convergencia nominal (generaciones máximas para el cuerpo del micro-AG). El límite inferior de este parámetro es dos, y es recomendable que no se exceda de seis.

int m Representa la cantidad de individuos en la población de trabajo del micro-AG. Este parámetro debe de ser mayor o igual a cuatro. Únicamente toma valores múltiplos de dos. Es recomendable usar cuatro en este parámetro.

double pc Indica qué porcentaje de la población habrá de cruzarse en cada ciclo evolutivo (conocido como porcentaje de cruza). Este parámetro es normalmente colocado en 0.7.

double pm El porcentaje de mutación indica la frecuencia con que se mutará cada uno de los alelos, pertenecientes a los genes de cada individuo. Su valor generalmente esta dado por la relación $1/bits\ del\ cromosoma$.

```
function Micro-AG
  begin
    Generar la población inicial P de tamaño N
    y guardar su contenido en la memoria de población M
    /* Ambas porciones de M seran llenadas con soluciones aleatorias */
    i=0
    while i < Max do
      begin
        Obtenerla población de trabajo (P^t) de M
          repeat
            begin
              Aplicar selección por torneo binario
              basado en no dominancia
              Aplicar cruza de dos puntos y mutación uniforme
              a los individuos seleccionados
              Aplicar elitismo (reteniendo solamente uno
              de los vectores no dominados)
              Producir la próxima generación
            end
          until convergencia nominal es alcanzada
        Malla-adaptativa(vecotres no dominados de P^t
        Copiar dos vectores no dominados de P^t a M
        if i mod ciclo de reemplazo
        then aplicarla segunda forma de elitismo
        i = i + 1
      end while
  end function
```

Figura 3.12: Pseudo-código del Micro-AG.

int ngenes Indica el número de genes (variables de decisión) con que el sistema trabajará.

int nobjs Representa el total de objetivos con que el sistema trabajará.

int tampareto Tamaño deseado del conjunto de óptimos de Pareto que debería obtener el sistema.

int tammem El tamaño de la memoria de población generalmente se establece a la mitad de *tampareto*.

int ndivs Representa el número de subdivisiones que se desea tenga la malla adaptativa. El rango de este parámetro va de 15 a 30 de acuerdo al resultado deseado del frente de Pareto (a mayor número más extendidas estarán las soluciones; sin embargo el algoritmo será también más lento). Un valor aceptable es 25.

int b Indica en qué base se requiere que éste la representación. Generalmente se utiliza una representación binaria ($b = 2$).

double *linf Establece el límite inferior de cada una de las variables. Si todas las variables con que el sistema trabajará son iguales, entonces únicamente es necesario establecer el valor correspondiente a la primera variable.

double *lsup Establece el límite superior de cada una de las variables. Si todas las variables con que el sistema trabajará son iguales, entonces únicamente es necesario establecer el valor correspondiente a la primera variable.

double *pres Establece la precisión requerida por cada variable.

int sndelit Representa el ciclo de reemplazo (o segundo elitismo), explicado en *Filtro*. El valor recomendado para este parámetro depende de *itot*, pero es deseable que sea de 10.

double pNRem Establece qué cantidad de la memoria de población será *no reemplazable*. Este parámetro generalmente adopta un valor de 0.3.

Capítulo 4

Funciones de prueba y análisis de resultados

4.1 Introducción

Para validar el desempeño del algoritmo propuesto, se evaluó su comportamiento en seis funciones de prueba tomadas de la literatura especializada y se compararon los resultados obtenidos contra los producidos por dos técnicas recientes: el *NSGA II* y *PAES*. Para realizar una comparación cuantitativa, se utilizaron cuatro métricas, de las cuales tres fueron propuestas previamente. Debido a que algunas de estas métricas requieren el verdadero frente de Pareto, éste se generó mediante enumeración exhaustiva (con una cierta granuralidad relativamente gruesa).

En este capítulo se introducen las métricas y se da una breve descripción de las técnicas rivales. Posteriormente, se describen las funciones de prueba y se presentan los resultados obtenidos por cada algoritmo al aplicar cada una de las métricas. Finalmente, se realiza un análisis de estos resultados y se discuten brevemente.

4.2 Métricas y contendientes

4.2.1 Métrica de la tasa de error (ER)

Esta métrica fue seleccionada para medir qué porcentaje del frente de Pareto obtenido (FP_o) no pertenece al frente de Pareto verdadero (FP_v).

$$ER = \frac{\sum_{i=1}^{n} e_i}{n} \tag{4.1}$$

donde:

n es el número de vectores en FP_o y

$$e_i = \begin{cases} 0 & \text{if vector } i \in FP_v,\ i = (1, \ldots n) \\ 1 & \text{en otro caso.} \end{cases}$$

Esta métrica muestra que si la corrida convergió o no a FP_v. Mientras más se aproxime ER a cero significa que tiene una mejor traslapación con FP_v. Una clara desventaja de esta métrica es la ausencia de la medición del extendimento de FP_o sobre FP_v.

4.2.2 Métrica del espacio distribuido (S)

Esta métrica nos muestra que tan bien se ha distribuido FP_o sobre el espacio que ocupa. Su definición matemática es la siguiente:

$$S = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\bar{d} - d_i)^2} \tag{4.2}$$

donde :

$d_i = min(\mid f_1^i(\bar{x}) \ - \ f_1^j(\bar{x}) \mid) \ + \ min(\mid f_2^i(\bar{x}) \ - \ f_2^j(\bar{x}) \mid),\ i, j = 1, \ldots, n,$

$\bar{d}$ es el promedio de todas las n, y

n es el número de vectores en FP_o.

Un valor de cero en esta métrica significa que todos los miembros de FP_0 son equidistantes espacialmente. Esta métrica, además de presentar el problema que presenta la métrica anterior, puede también caer en otro error más grave al no incluir dentro de su definición a FP_v, ya que existe la posibilidad que cualquier corrida (aún una alejada del frente de Pareto) obtenga una puntación sobresaliente en esta métrica.

4.2.3 Métrica M1

Esta métrica fue propuesta por Zitzler et al [73]. Genera como resultado la distancia promedio a la que se encuentra FP_o de FP_v. Su definición matemática se muestra a continuación:

$$M1(X') = \frac{1}{\mid X' \mid} \sum_{\overline{a} \in R'} min \| \ a' - \overline{a} \ \| ; \overline{a} \in \overline{X} \tag{4.3}$$

- donde:

- $X', \overline{X} \subset X$ son conjuntos de vectores objetivo, siendo X el conjunto correspondiente al frente de Pareto real.

Si $M1$ tuviera un valor de cero significaría que FP_o es un subconjunto de FP_v. Esta métrica también presenta el mismo inconveniente que sus antecesoras, ya que no toma en cuenta la distribución a lo largo del frente de Pareto.

4.2.4 Métrica O

La métrica "O" está basada en la M_1 de Zitzler [73] y fue desarrollada para incluir vectores solución de diferentes corridas, con el objetivo de realizar una comparación directa entre ellos y generando un valor para cada métrica. El valor más pequeño de la aplicación de esta métrica será para aquella corrida que haya abarcado una porción más extensa de FP_v y que esté más cerca de éste. La métrica O se define de la siguiente manera:

Sea a'_i, sea $\overline{a}_{mi} \in \overline{X}$ tal que $|| \, a' - \overline{a}_{mi} \, || = min|| \, a' - \overline{a} \, ||; \overline{a} \in \overline{X}$

$$M(X', P) = \frac{1}{| \, X' \, |} \sum_{a'_i \in X' ||, \overline{a} \in \overline{X}} \tag{4.4}$$

$$R = \overline{a}_{mi}{}_{i=1}^{|x'|} \tag{4.5}$$

$$M^*(R', X'') = \frac{1}{| \, X' \, |} \sum_{\overline{a}_m \in R'} min|| \, a_m - a' \, ||; a' \in X'' \tag{4.6}$$

$O_1 = M(X', P) + \sum_{i=2}^{n} M^*(R^{i'}, X')$
$O_2 = M(X'', P) + \sum_{i=2}^{n} M^*(R^{i'}, X'')$
$O_3 = M(X''', P) + \sum_{i=2}^{n} M^*(R^{i'}, X''')$
$O_n = M(X^{n'}, P) + \sum_{i=2}^{n} M^*(R^{i'}, X^{n'})$

Donde: R', R'', R''', Rn' son vectores de soluciones correspondientes X', X'', X''', Xn' y n es el número de vectores de solución. En otras palabras, para cada solución de X' busca la solución más cercana a ella.

4.2.5 Contendientes

Se revisó el estado del arte en la disciplina con el fin de identificar algoritmos evolutivos que tuvieran un desempeño robusto y que representaran las tendencias actuales de investigación en esta área.

La primera técnica seleccionada fue PAES [37], debido a su velocidad, manejo eficiente de las soluciones no dominadas (mediante una malla adaptativa) y simplicidad. La segunda técnica elegida fue el NSGA II [18] por su alto desempeño y sus excelentes capacidades de explotación.

4.2.5.1 PAES

PAES es una técnica propuesta reciente por *Knowles y Corne* [37]. Se trata de una estrategia evolutiva en su forma más simple *(1+1)-EE* (un padre genera un único hijo), que junto a un archivo histórico de soluciones no dominadas y un mecanismo para mantener diversidad genera un conjunto de soluciones no dominadas eficientemente distribuidas a lo largo del frente de Pareto. La malla adaptativa es el mecanismo que trata de mantener la diversidad en las soluciones no dominadas mediante la división recursiva del espacio de búsqueda (generando regiones geográficas de búsqueda). Debido a que el proceso para mantener diversidad es adaptativo, no son necesarios parámetros extra. Esto es sin duda una clara ventaja sobre las técnicas que utilizan nichos [19].

El pseudocódigo de PAES [37] se muestra en la figura 4.1.

```
begin
  Generar la solución aleatoria c y añadirla al archivo
  repeat
    mutar c para para producir m y evaluar m
    if (c domina a m ) descartar m
    else if (m domina a c)
      remplazar c por m, y añadir m a el archivo
    else if (m es dominada por cualquier miembro del archivo)
        descartar m
    else aplicar malla(c,m,archive) para determinar cual va a
        ser la nueva solución actual y si se añadirá m al archivo
  until un criterio de terminación sea alcanzado
end
```

Figura 4.1: Pseudocódigo del PAES

4.2.5.2 NSGA II

El NSGA II fue desarrollado por *Deb et al* [18] con el propósito de mejorar el desempeño de procesos que generalmente son objeto de críticas en los algoritmos evolutivos y claramente visibles en la versión original del NSGA: el proceso de ordenar la población basado en no dominancia $O(mN^3)$ (donde m es igual a la cantidad de objetivos, y N es el tamaño de la población), el uso de técnicas carentes de elitismo y el uso de parámetros extra tales como el parámetro de compartición de aptitud.

El NSGA II es un algoritmo altamente elitista con un desempeño robusto para problemas con o sin restricciones. El pseudocódigo de esta técnica está plasmado en la figura 4.2.

begin
  Generar `aleatoriamente la población inicial,` $G(0)$
  sea $t := 0$
  repeat
    calcular `la aptitud de cada individuo de` $G(t)$
    asignar `compartición de aptitud`
    computar `el contador de nichos`
    seleccionar $G_1(t)$ `mediante selección estocástica de` $G(t)$
    aplicar `los operadores genéticos a` $G_1(t)$ `para generar` $G(t+1)$
    sea $t := t+1$
  until `que cierta condición se satisfaga`
end

Figura 4.2: Pseudocódigo del NSGA II

4.3 Parámetros utilizados en las técnicas

Para efecto de realizar una adecuada medición del desempeño, los algoritmos comparados para efectos de la validación del micro-AG se ejecutaron un mismo número de evaluaciones para la función de aptitud en cada problema de prueba (efectuándose un total de veinte corridas de cada algoritmo).

A continuación se muestran los valores que tomaron los parámetros utilizados por cada algoritmo:

Micro-GA
Tamaño del archivo externo = 100
Tamaño de la memoria de población = 50
Porcentaje no reemplazable de la memoria de población = 0.3
Porcentaje de cruza = 0.7
Porcentaje de mutación = 1/bits del cromosoma
Número de iteraciones para alcanzar convergencia nominal= 2
Número de subdivisiones de la malla adaptativa = 25
Ciclo del segundo elitismo =50
Tamaño de población = 4

NSGA II
Tamaño de población = 100
Porcentaje de mutación = 1/vars, donde vars = número de las variables de decisión del problema
Porcentaje de cruza = 0.8 (usando SBX)
selección por torneo

PAES
Tamaño del archivo = 100
Profundidad de la malla adaptativa = 5
Porcentaje de mutación = 1/bits del cromosoma

Si algún parámetro tomara otro valor, éste será expresado en el problema en que esto ocurra.

4.4 MOP 1

El primer problema abordado es una función de dos objetivos propuesta por Deb [17]:

$$\text{Minimizar } f_1(x_1, x_2) = x_1 \tag{4.7}$$

$$\text{Minimizar } f_2(x_1, x_2) = \frac{g(x_2)}{x_1} \tag{4.8}$$

donde:

$$g(x_2) = 2.0 - \exp\left\{-\left(\frac{x_2 - 0.2}{0.004}\right)^2\right\} - 0.8\exp\left\{-\left(\frac{x_2 - 0.6}{0.4}\right)^2\right\} \tag{4.9}$$

y $0.1 \leq x_1 \leq 1.0$, $0.1 \leq x_2 \leq 1.0$.

Este problema tiene un frente de Pareto local que sirve de atracción para la mayoría de las soluciones. Para poder observar mejor la complejidad de este problema, se generaron 20,000 soluciones aleatorias. La gráfica de estas soluciones y del verdadero frente de Pareto se muestran en la figura 4.3.

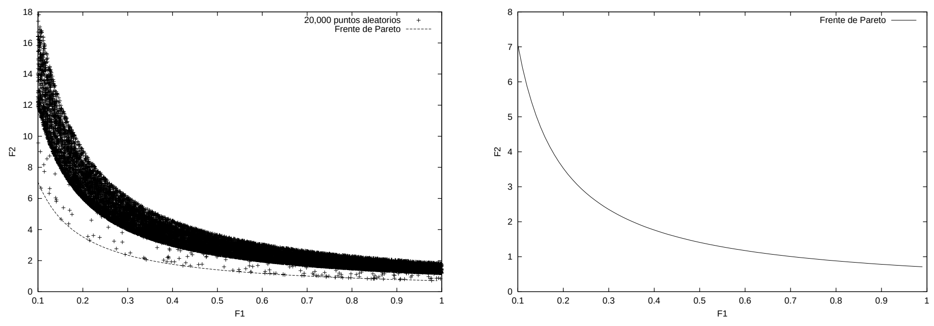


Figura 4.3: La gráfica de la izquierda muestra 20,000 soluciones generadas aleatoriamente y la derecha su frente de Pareto verdadero (MOP 1).

El número de evaluaciones de la función de aptitud que realizaron los tres algoritmos en cada una de las 20 corridas para este problema fue de 12,000. El micro-AG tuvo un tiempo promedio de ejecución en las corridas de 0.20 segundos, el NSGA II de 2.601 segundos, y el PAES de 0.5 segundos.

Los parámetros modificados por algoritmo son enumerados a continuación:

- Micro-AG:

  1. Número de iteraciones = 1,500.

2. Porcentaje de mutación = 0.029.

- NSGA II:

1. Porcentaje de mutación = 0.5.

- PAES:

1. Porcentaje de mutación = 0.029.

4.4.1 Resultados de las métricas aplicadas

4.4.1.1 Métrica de la tasa de error

En la tabla 4.1 se pueden observar las estadísticas obtenidas de la aplicación de la métrica ER a veinte corridas de cada uno de los algoritmos. En la figura 4.4 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	Micro-AG	NSGA II	PAES
Mejor	4	0	2
Peor	24	101	101
Promedio	12.3	16.6	26.8
Desviación Estándar	7.392029847	36.299050299	33.24328187
Varianza	54.642105263	1317.621052632	1105.115789474

Tabla 4.1: Tabla de estadísticas de ER (MOP 1).

4.4.1.2 Métrica espacio distribuido

En la tabla 4.2 se pueden observar las estadísticas obtenidas de la aplicación de la métrica denominada espacio distribuido a veinte corridas de cada uno de los algoritmos y en la figura 4.5 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	Micro-AG	NSGA II	PAES
Mejor	0.0261238	0.0302648	0.0215848
Peor	0.0426045	0.0682254	0.0434856
Promedio	0.03195151	0.03719832	0.030784285
Desviación Estándar	0.006140425	0.010237417	0.006891366
Varianza	0.000037705	0.000104805	0.000047491

Tabla 4.2: Tabla de estadísticas de la métrica espacio distribuido (MOP 1).

4.4.1.3 Métrica M1

En la tabla 4.3 se pueden observar las estadísticas obtenidas de la aplicación de la métrica $M1$ a veinte corridas de cada uno de los algoritmos y en la figura 4.6 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mejor	0.0222318	0.0203253	0.0219157
Peor	0.282308	0.973059	1.09936
Promedio	0.076460175	0.184082665	0.17491824
Desviación estándar	0.0776446085336182	0.335559127127996	0.25193671676105
Varianza	0.00602868523433882	0.112599927798902	0.0634721092523373

Tabla 4.3: Tabla de estadísticas de M1 (MOP 1).

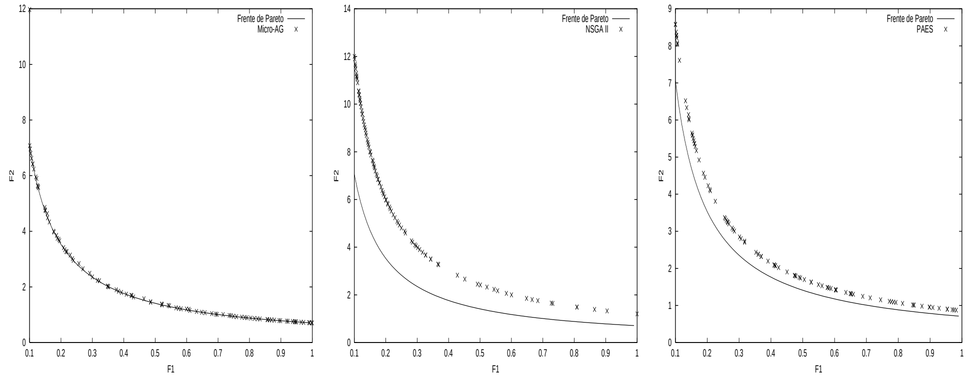
4.4.1.4 Métrica 0

En la tabla 4.4 se pueden observar las estadísticas obtenidas de la aplicación de la métrica O a veinte corridas de cada uno de los algoritmos. En la figura 4.7 se grafican las peores, medianas y mejores corridas por algoritmo.

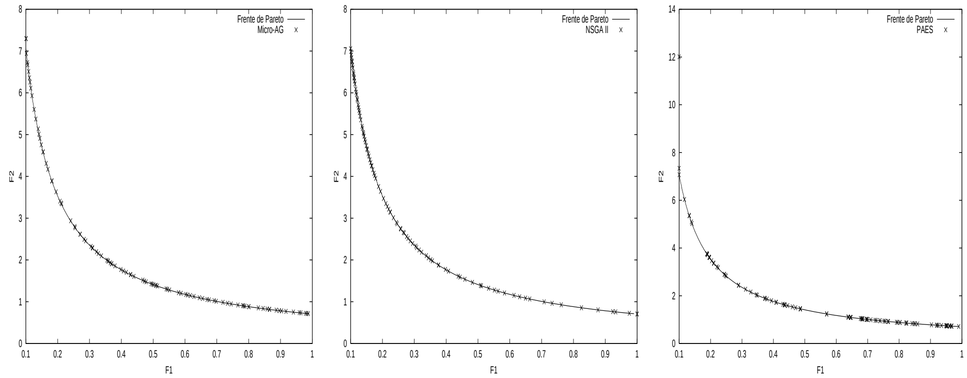
Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mejor	0.0262398	0.0296641	0.047201
Peor	0.127233	0.560979	0.479401
Promedio	0.057627515	0.119467765	0.136393185
Desviación Estándar	0.030878957	0.187238408	0.112528329
Varianza	0.00095351	0.035058221	0.012662625

Tabla 4.4: Tabla de estadísticas de métrica O (MOP 1).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas

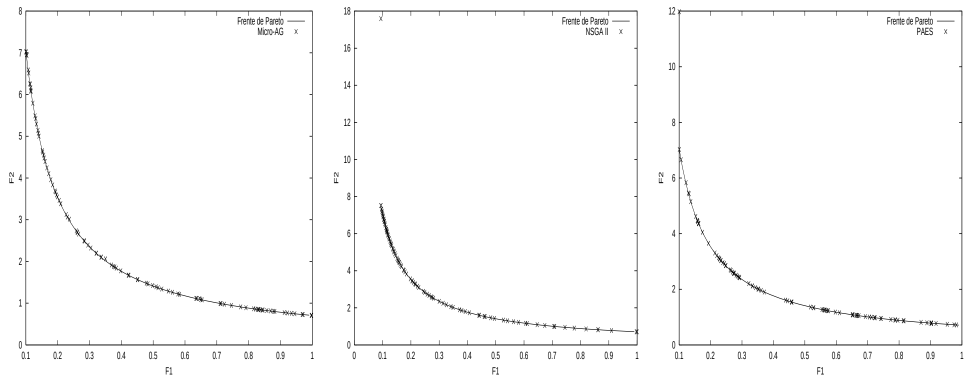
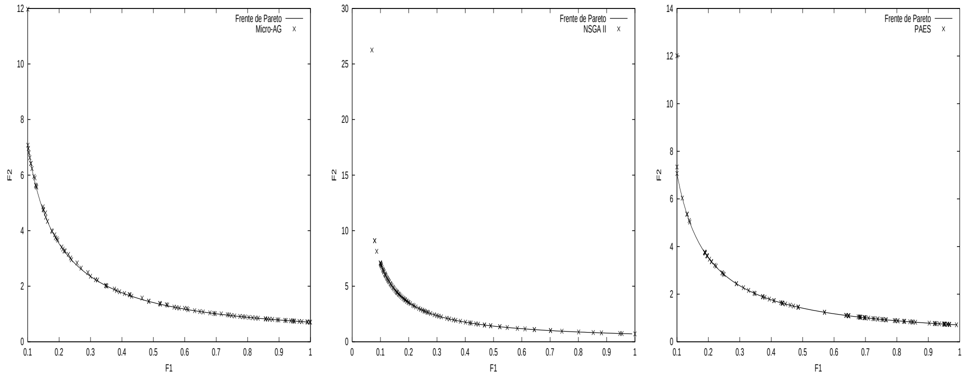
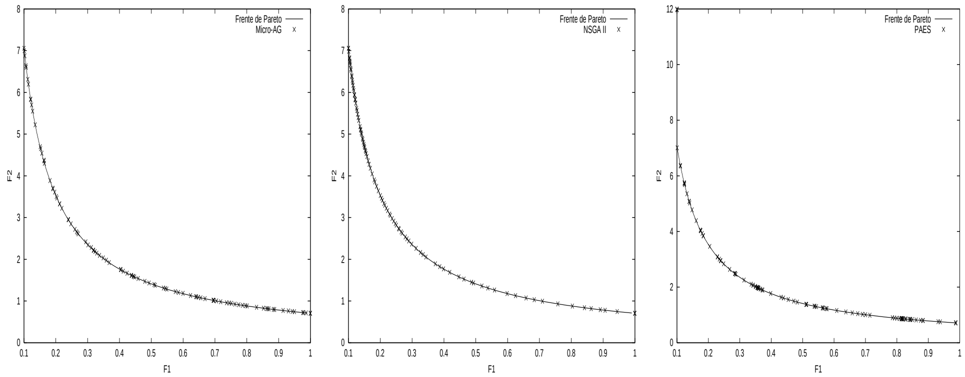


Figura 4.4: Gráficas de las peores, medianas y mejores corridas de acuerdo a la métrica *ER* (MOP1).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

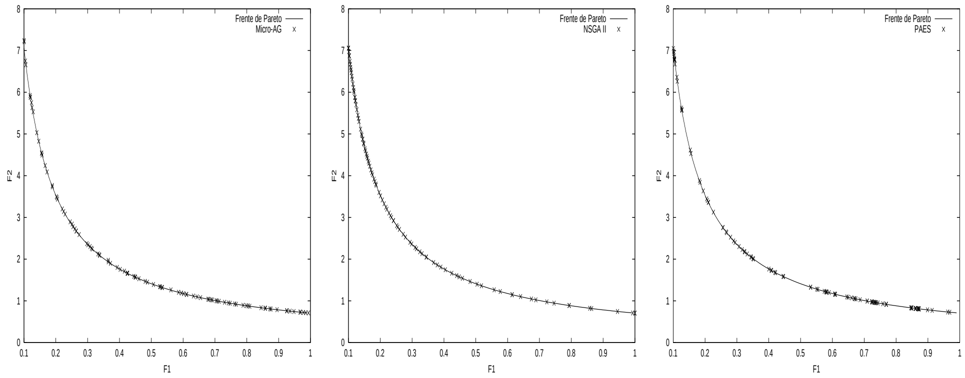
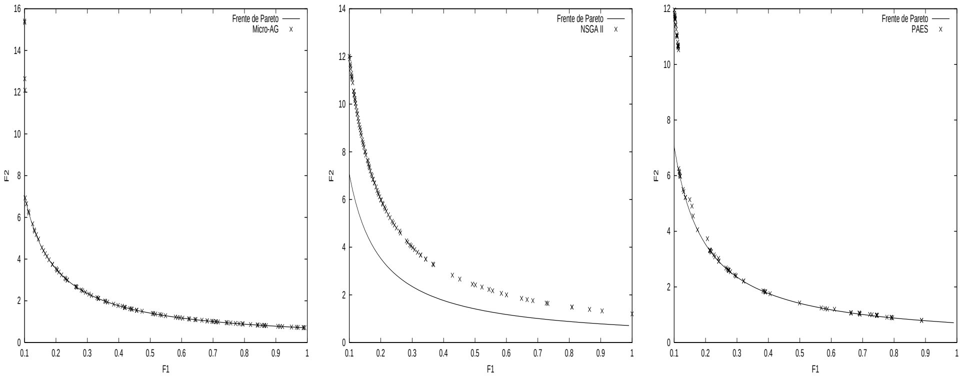
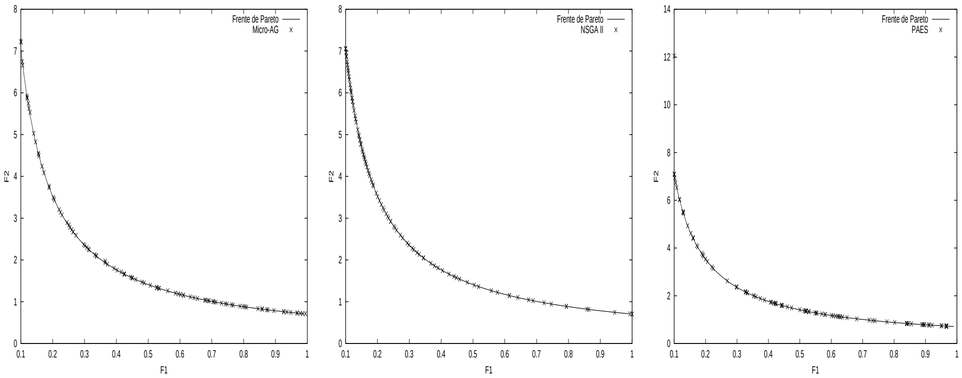


Figura 4.5: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica de *espacio distribuido* (MOP1).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

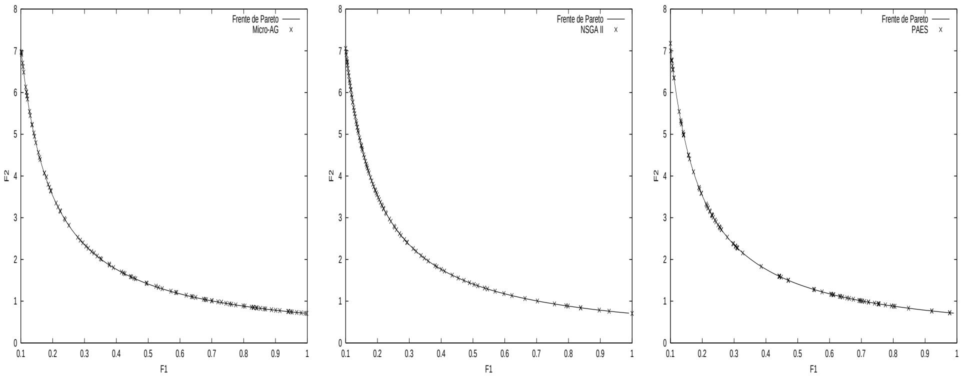
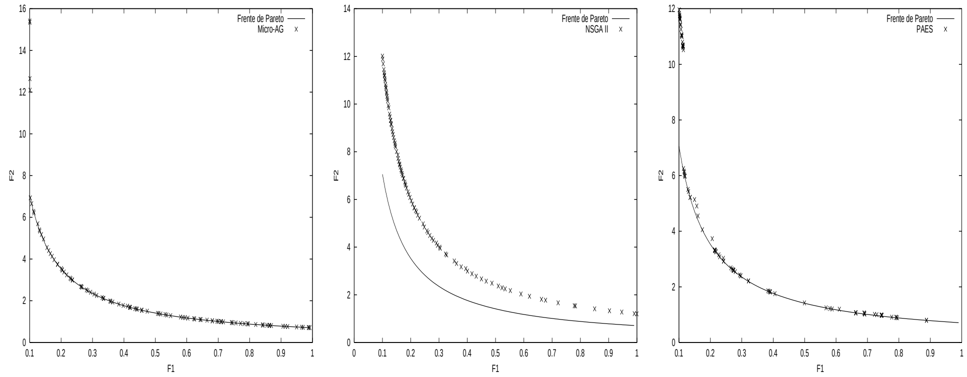
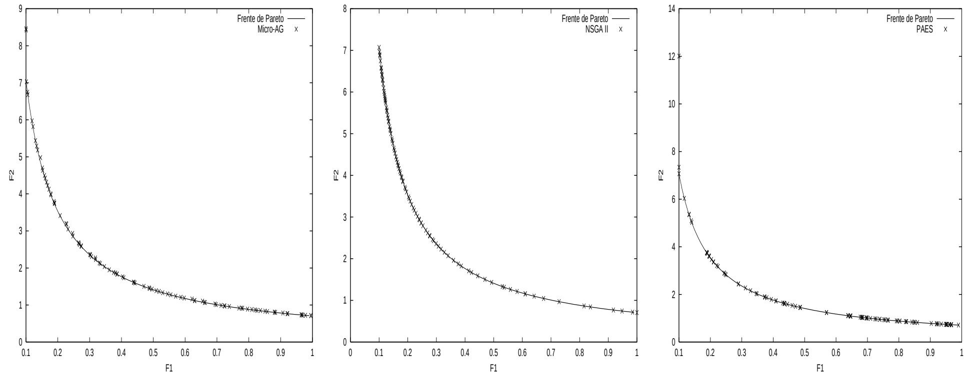


Figura 4.6: Frente de Pareto correspondientes a las peores,medianas y mejores corridas de acuerdo a la métrica $M1$ (MOP1).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

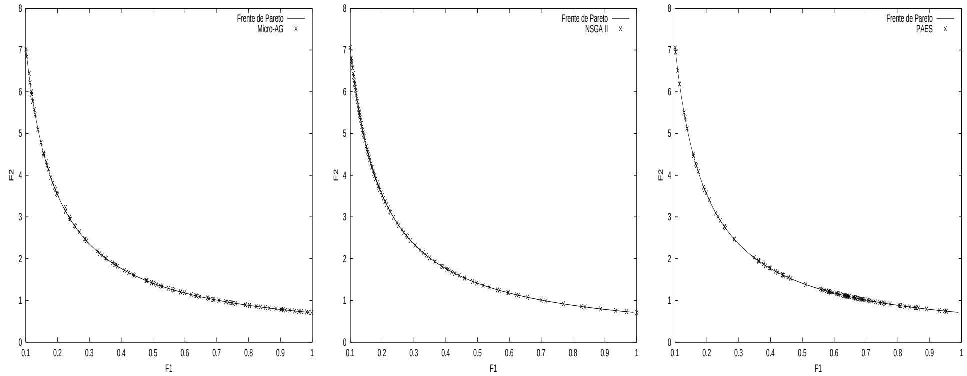


Figura 4.7: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica O (MOP1).

4.5 MOP 2

El segundo ejemplo es un problema de optimización de dos objetivos propuesto por Schaffer [58] que ha sido usada por diferentes investigadores [62, 6]:

$$\text{Minimizar } f_1(x) = \begin{cases} -x & \text{if } x \leq 1 \\ -2+x & \text{if } 1 < x \leq 3 \\ 4-x & \text{if } 3 < x \leq 4 \\ -4+x & \text{if } x > 4 \end{cases} \tag{4.10}$$

$$\text{Minimizar } f_2(x) = (x-5)^2 \tag{4.11}$$

y $-5 \leq x \leq 10$.

Conjunto óptimo de Pareto desconectado. Frentes de Pareto desconectado

Este problema presenta tanto el conjunto óptimo de Pareto como el frente de Pareto desconectados. La figura 4.8 es presentada para mostrar la complejidad de este problema.

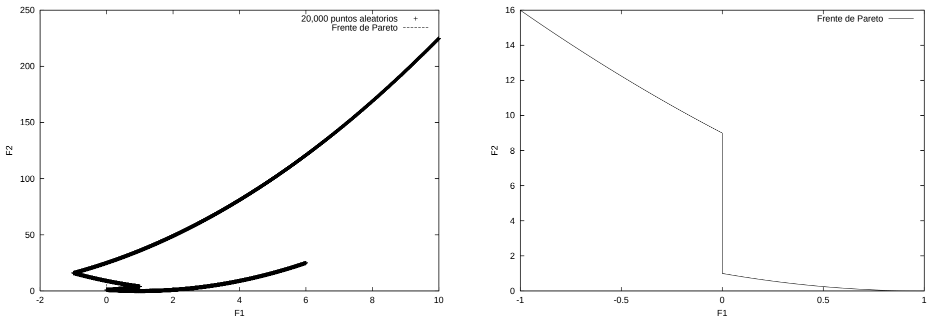


Figura 4.8: La gráfica de la izquierda muestra 20,000 soluciones generadas aleatoriamente y la derecha su frente de Pareto verdadero (MOP 2).

El número de evaluaciones de la función de aptitud que realizaron los tres algoritmos en cada una de las 20 corridas para este problema fue de 1,200. El micro-AG tuvo un tiempo promedio de ejecución en las corridas de 0.017 segundos, el NSGA II de 0.282 segundos, y el PAES de 0.017 segundos.

Los parámetros modificados por algoritmo son enumerados a continuación:

- Micro-AG:
  1. Número de iteraciones = 150.
  2. Porcentaje de mutación = 0.056.
- NSGA II:
  1. Porcentaje de mutación = 0.5.
- PAES:
  1. Porcentaje de mutación = 0.056.

4.5.1 Resultados de las métricas aplicadas

4.5.1.1 Métrica de la tasa de error

En la tabla 4.5 se pueden observar las estadísticas obtenidas de la aplicación de la métrica ER a veinte corridas de cada uno de los algoritmos. En la figura 4.9 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	Micro-AG	NSGA II	PAES
Mínimo	0	0	0
Máximo	4	6	31
Promedio	1.65	2.5	7.75
Desviación Estándar	0.988086934	1.432700799	11.737815631
Varianza	0.976315789	2.052631579	137.776315789

Tabla 4.5: Tabla de estadísticas de ER (MOP 2).

4.5.1.2 Métrica espacio distribuido

En la tabla 4.6 se pueden observar las estadísticas obtenidas de la aplicación de la métrica denominada espacio distribuido a veinte corridas de cada uno de los algoritmos y en la figura 4.10 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	Micro-AG	NSGA II	PAES
Mínimo	0.0260238	0.0373541	0.01892
Máximo	0.0335996	0.0400734	0.0661497
Promedio	0.029883435	0.038894945	0.028992495
Desviación Estándar	0.001993682	0.000754153	0.010521584
Varianza	0.000003975	0.000000569	0.000110704

Tabla 4.6: Tabla de estadísticas de la métrica de espacio distribuido (MOP 2).

4.5.1.3 Métrica M1

En la tabla 4.7 se pueden observar las estadísticas obtenidas de la aplicación de la métrica $M1$ a veinte corridas de cada uno de los algoritmos y en la figura 4.11 se grafican las peores, medianas y mejores corridas por algoritmo.

4.5.1.4 Métrica 0

En la tabla 4.8 se pueden observar las estadísticas obtenidas de la aplicación de la métrica O a veinte corridas de cada uno de los algoritmos. En la figura 4.12 se

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.000888145	0.00142107	0.000546909
Máximo	0.00362861	0.00192145	0.485627
Promedio	0.00153016236842105	0.00161422	0.0675201338
Desviación estándar	0.000767204753631955	0.00014151098105057	0.130317183309834
Varianza	5.88603133995468e-07	2.00253577578949e-08	0.0169825682658088

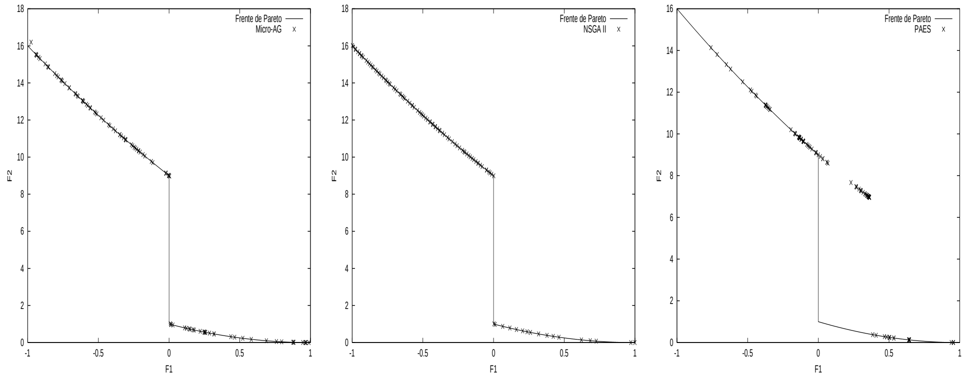
Tabla 4.7: Tabla de estadísticas de M1 (MOP 2).

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.0204262	0.013865	0.0677586
Máximo	0.0765601	0.0288934	3.95616
Promedio	0.03979466	0.01798264	0.41654333
Desviación Estándar	0.014051932	0.003146214	0.843951499
Varianza	0.000197457	0.000009899	0.712254133

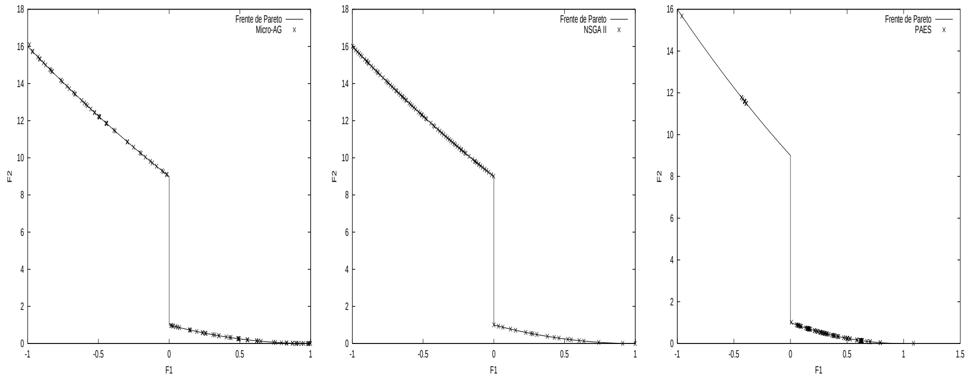
Tabla 4.8: Tabla de estadísticas de métrica O (MOP 2)

grafican las peores, medianas y mejores corridas por algoritmo.

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas

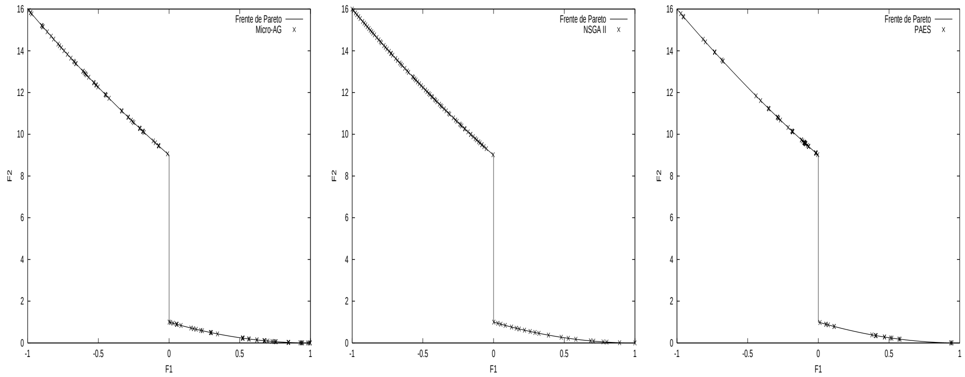
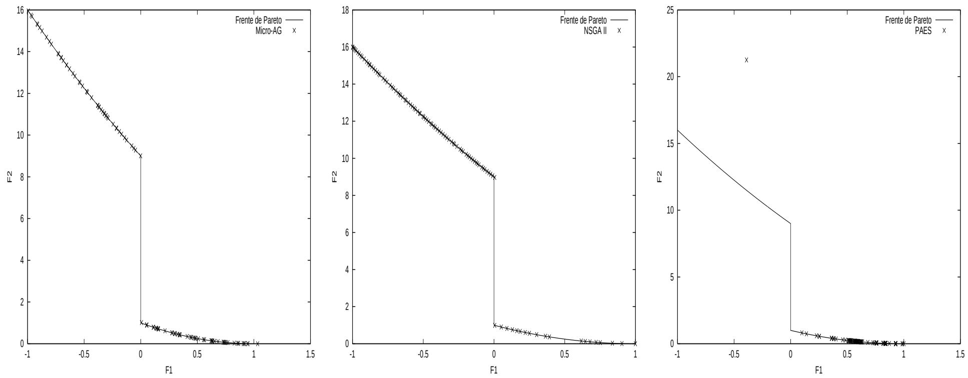
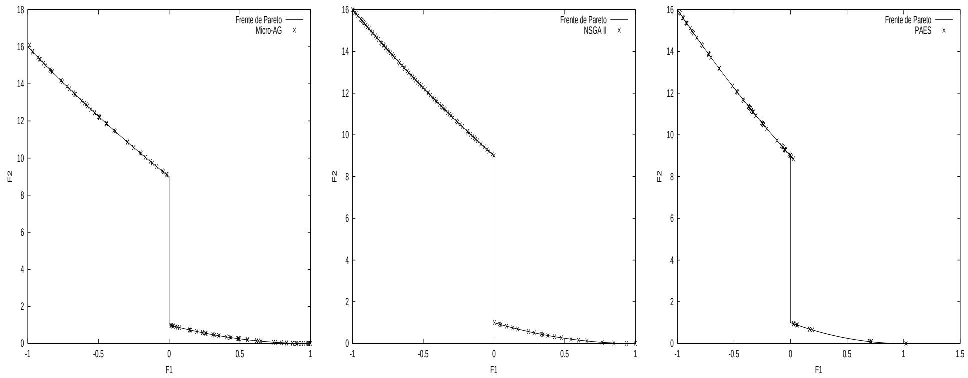


Figura 4.9: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica ER (MOP2).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

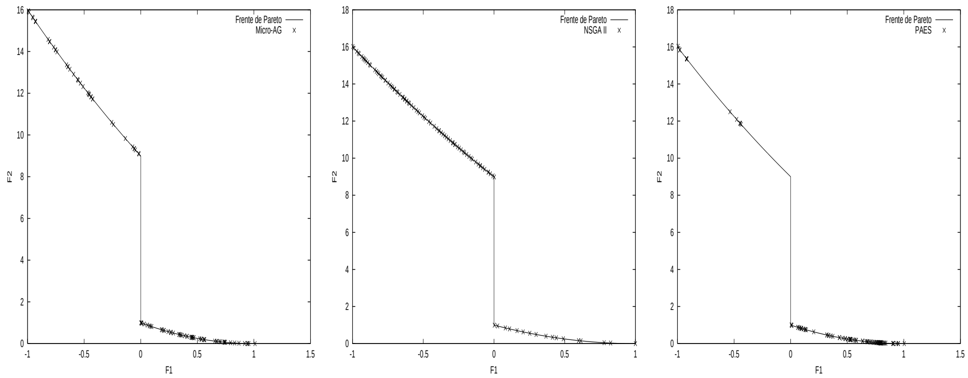
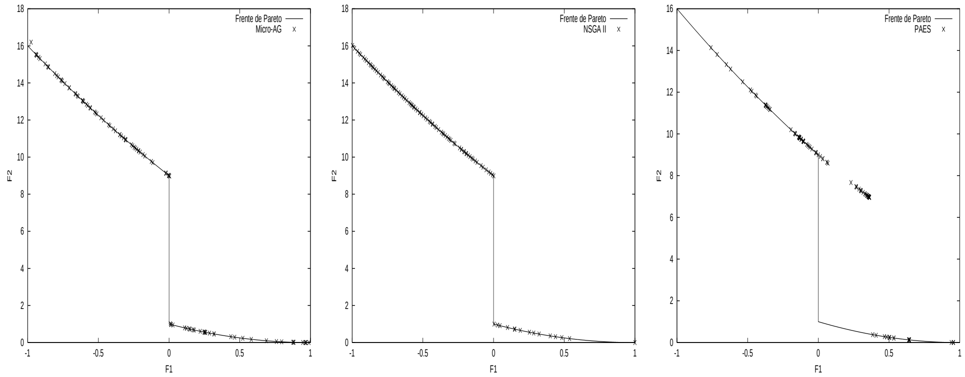
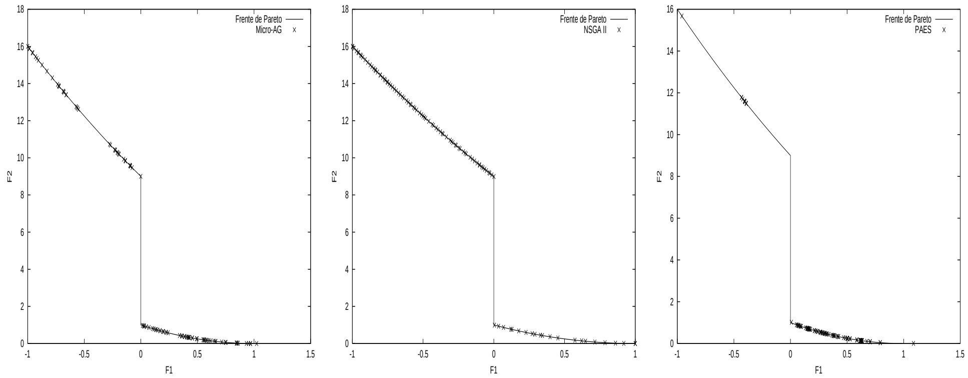


Figura 4.10: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica de *espacio distribuido* (MOP2).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

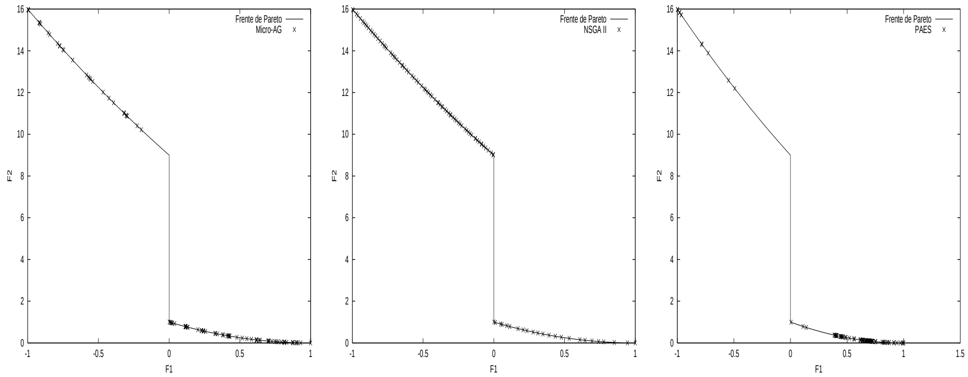
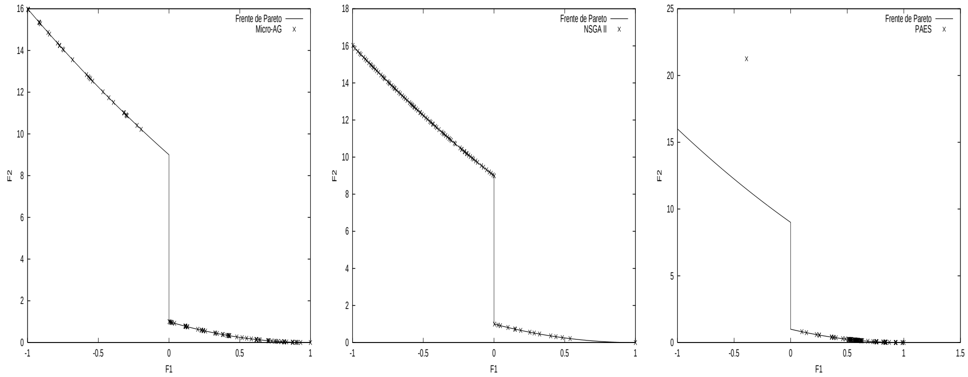
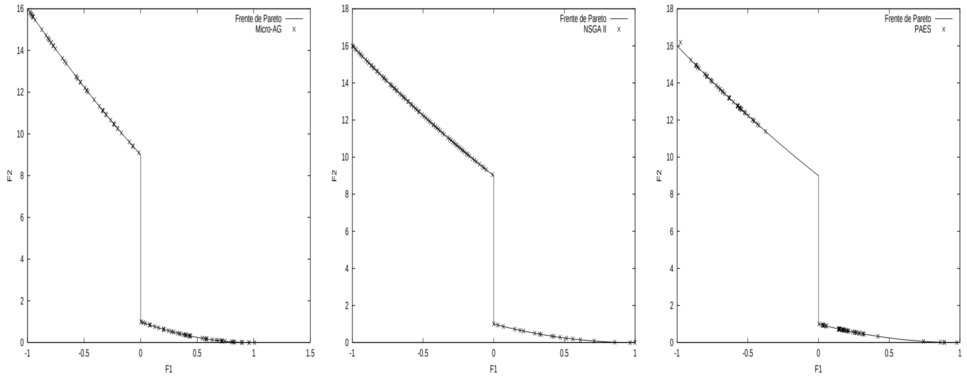


Figura 4.11: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica $M1$ (MOP2).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

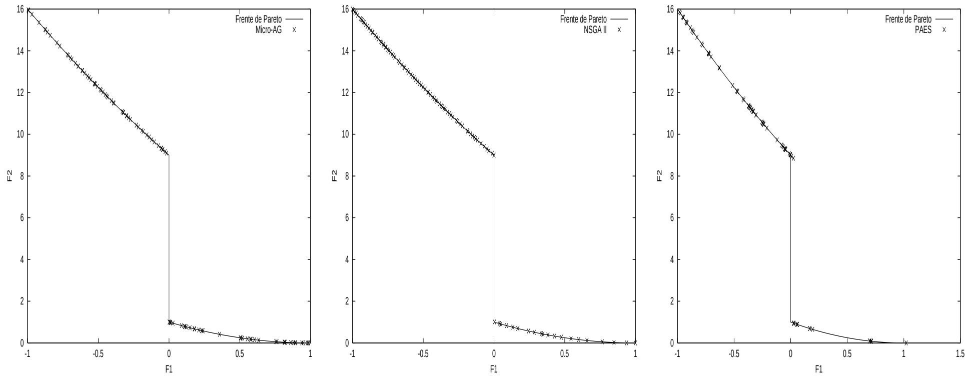


Figura 4.12: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica O (MOP2).

4.6 MOP 3

El tercer ejemplo es un problema optimización de dos objetivos propuesto por Deb [17]:

$$\text{Minimizar } f_1(x_1,x_2) = x_1 \tag{4.12}$$

$$\text{Minimizar } f_2(x_1,x_2) = g(x_1,x_2)\cdot h(x_1,x_2) \tag{4.13}$$

donde:

$$g(x_1,x_2) = 11 + x_2^2 - 10\cdot\cos(2\pi x_2) \tag{4.14}$$

$$h(x_1,x_2) = \begin{cases} 1 - \sqrt{\frac{f_1(x_1,x_2}{g(x_1,x_2}} & \text{if } f_1(x_1,x_2) \leq g(x_1,x_2) \\ 0 & \text{de lo contrario} \end{cases} \tag{4.15}$$

y $0 \leq x_1 \leq 1$, $-30 \leq x_2 \leq 30$.

La complejidad de este problema reside en la gran cantidad de frentes de Pareto locales (61). En la gráfica 4.13 es mostrada dicha complejidad mediante la generación de 20,000 soluciones aleatorias y la localización de su frente de Pareto global.

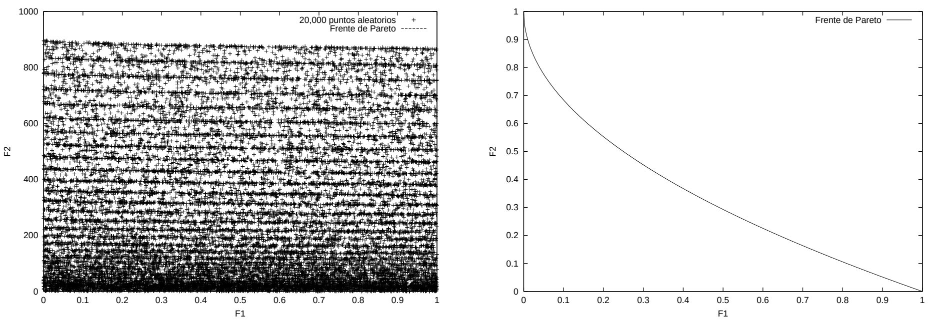


Figura 4.13: La gráfica de la izquierda muestra 20,000 soluciones generadas aleatoriamente y la derecha su frente de Pareto verdadero (MOP 3).

El número de evaluaciones de la función de aptitud que realizaron los tres algoritmos en cada una de las 20 corridas para este problema fue de 11,200. El micro-AG tuvo un tiempo promedio de ejecución en las corridas de 0.107 segundos, el NSGA II de 2.519 segundos, y el PAES de 2.497 segundos.

Los parámetros modificados por algoritmo son enumerados a continuación:

- Micro-AG:

  1. Número de iteraciones = 700.

2. Porcentaje de mutación = 0.029.
3. Número de iteraciones para alcanzar convergencia nominal = 4.

- NSGA II:

  1. Porcentaje de mutación = 0.5.

- PAES:

  1. Porcentaje de mutación = 0.029.

4.6.1 Resultados de las métricas aplicadas

4.6.1.1 Métrica de la tasa de error

En la tabla 4.9 se pueden observar las estadísticas obtenidas de la aplicación de la métrica *ER* a veinte corridas de cada uno de los algoritmos. En la figura 4.14 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	11	0	2
Máximo	72	85	101
Promedio	43.1	12.65	33.85
Desviación Estándar	22.289719131	19.618130712	39.966136982
Varianza	496.831578947	384.871052632	1597.292105263

Tabla 4.9: Tabla de estadísticas de ER (MOP 3).

4.6.1.2 Métrica espacio distribuido

En la tabla 4.10 se pueden observar las estadísticas obtenidas de la aplicación de la métrica denominada espacio distribuido a veinte corridas de cada uno de los algoritmos y en la figura 4.15 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.0132817	0.0159462	0.0136695
Máximo	0.0174982	0.0594295	0.297868
Promedio	0.01459075	0.01922441	0.04158688
Desviación Estándar	0.000915159	0.009874717	0.067302319
Varianza	0.000000838	0.00009751	0.004529602

Tabla 4.10: Tabla de estadísticas de la métrica de espacio distribuido (MOP 3).

4.6.1.3 Métrica M1

En la tabla 4.11 se pueden observar las estadísticas obtenidas de la aplicación de la métrica $M1$ a veinte corridas de cada uno de los algoritmos y en la figura 4.16 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.000831911	0.000660897	0.000802428
Máximo	0.00700336	0.16293	4.38637
Promedio	0.001729482	0.01045023525	0.52717412875
Desviación	0.00133948552973705	0.0364840255092269	1.03348518758406
estándar			
Varianza	1.79422148437495e-06	0.00133108411735792	1.06809163295565

Tabla 4.11: Tabla de estadísticas de M1 (MOP 3).

4.6.1.4 Métrica 0

En la tabla 4.12 se pueden observar las estadísticas obtenidas de la aplicación de la métrica O a veinte corridas de cada uno de los algoritmos. En la figura 4.17 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.00416133	0.00224422	0.00524734
Máximo	0.0129062	0.0567972	1.47304
Promedio	0.006364783	0.006431212	0.328798092
Desviación Estándar	0.002286411	0.01209414	0.491853365
Varianza	0.000005228	0.000146268	0.241919732

Tabla 4.12: Tabla de estadísticas de métrica O (MOP 3).

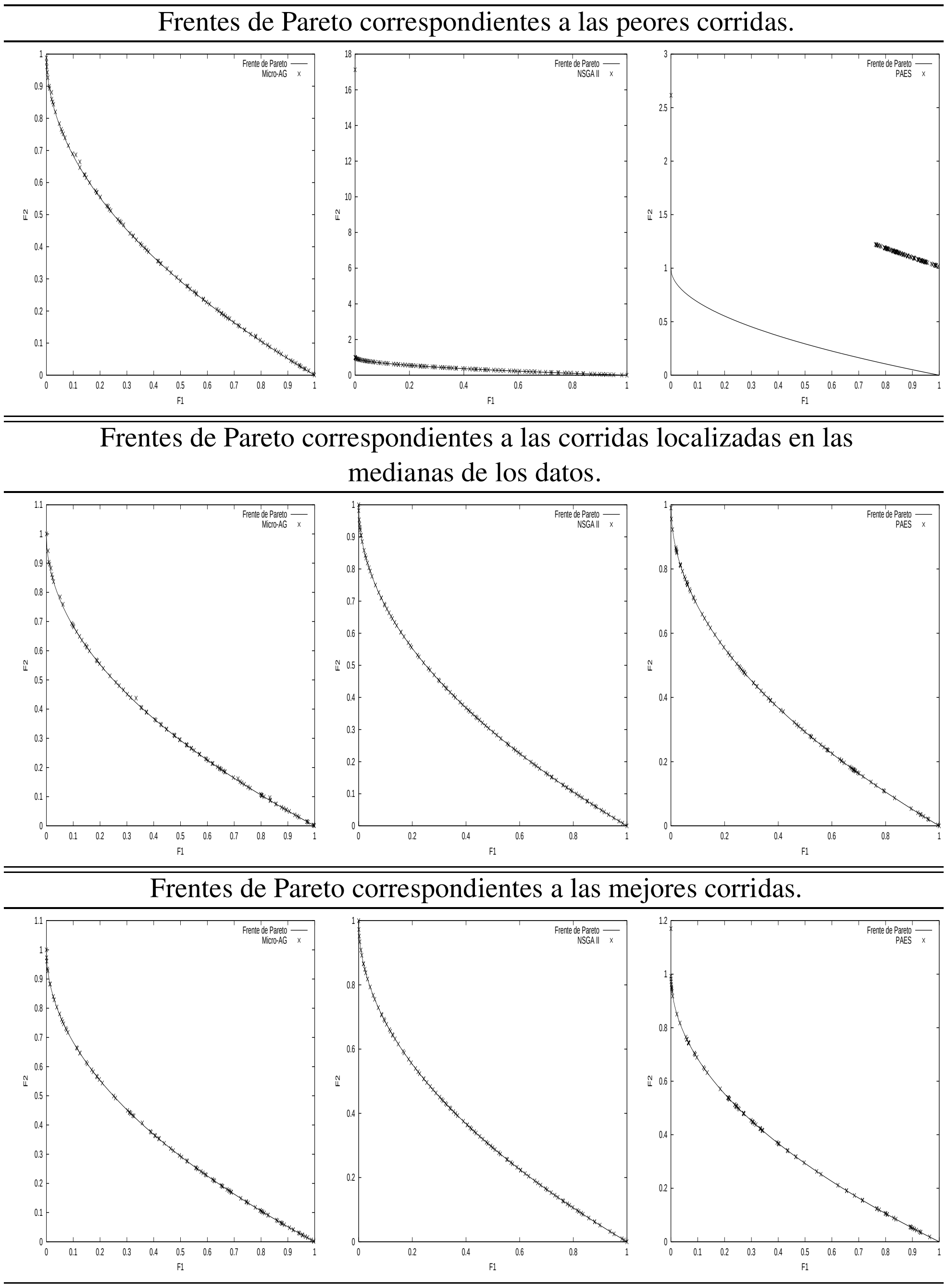
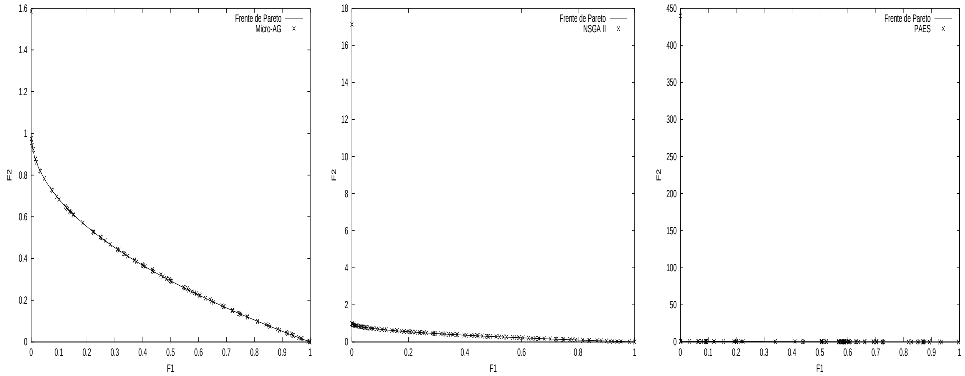
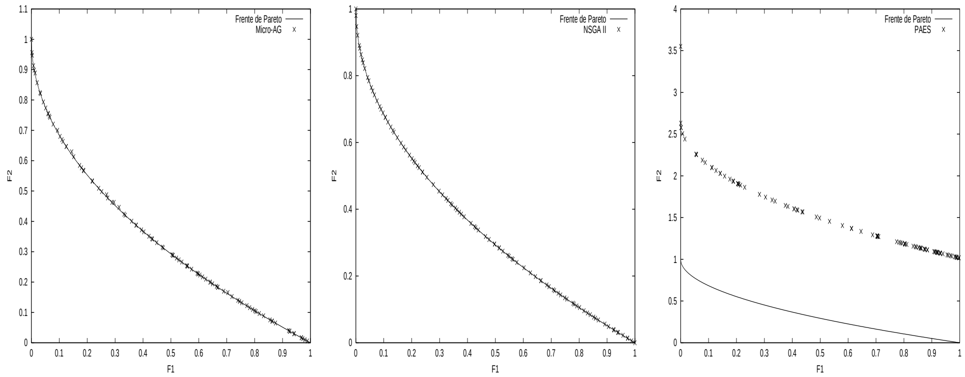


Figura 4.14: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas, de acuerdo a la métrica ER (MOP3).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

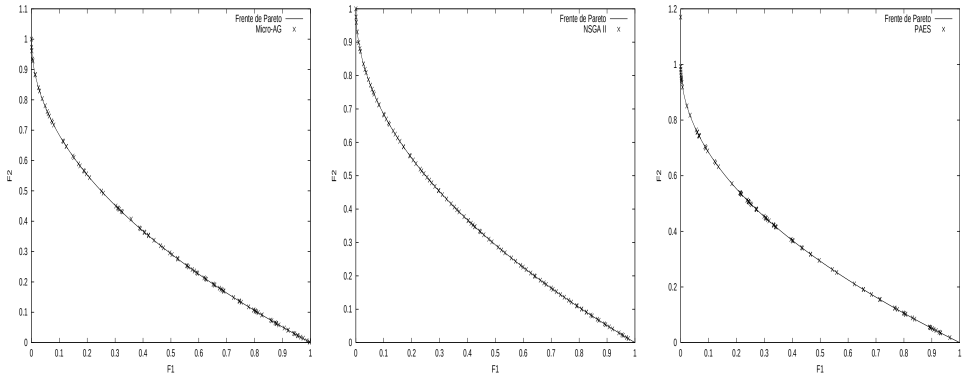
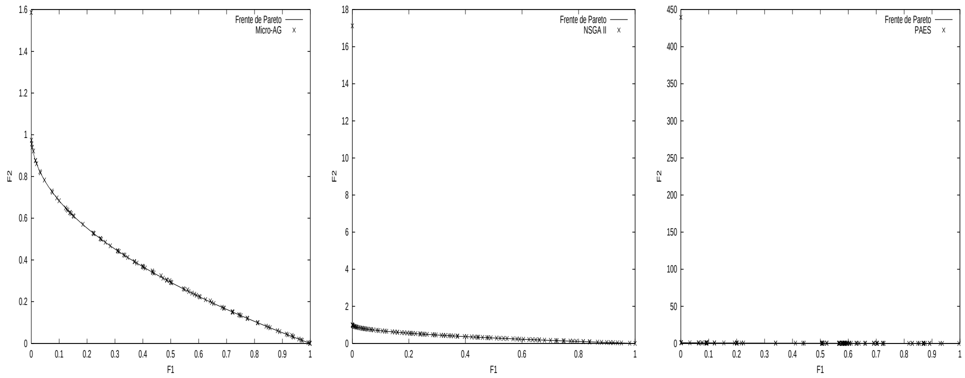
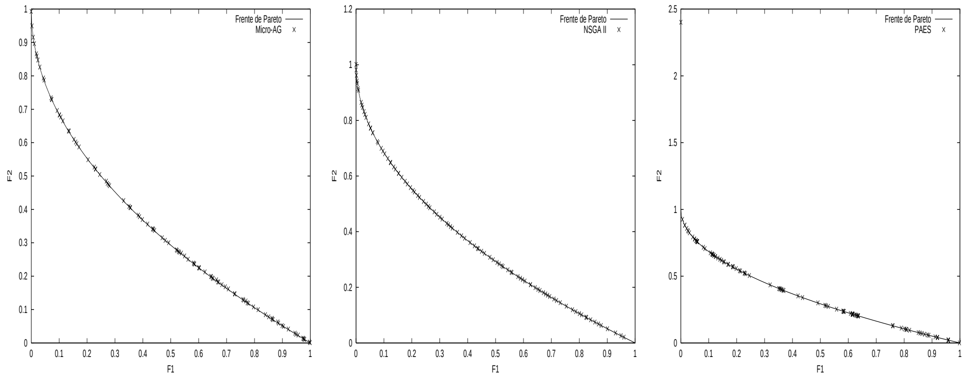


Figura 4.15: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica de *espacio distribuido* (MOP3).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

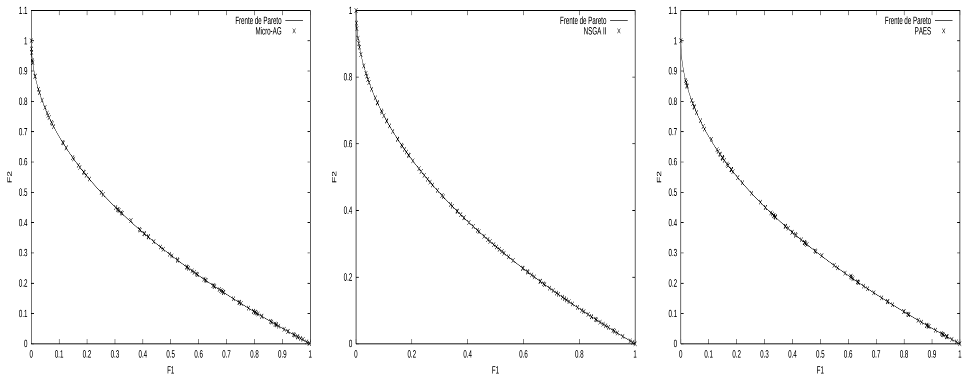
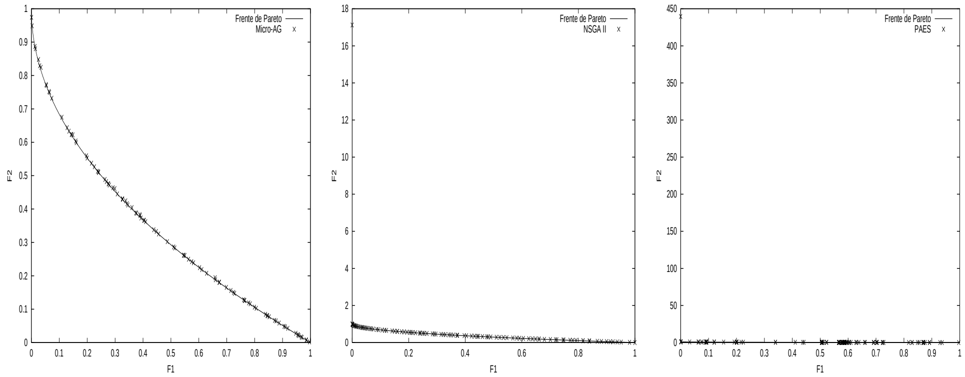
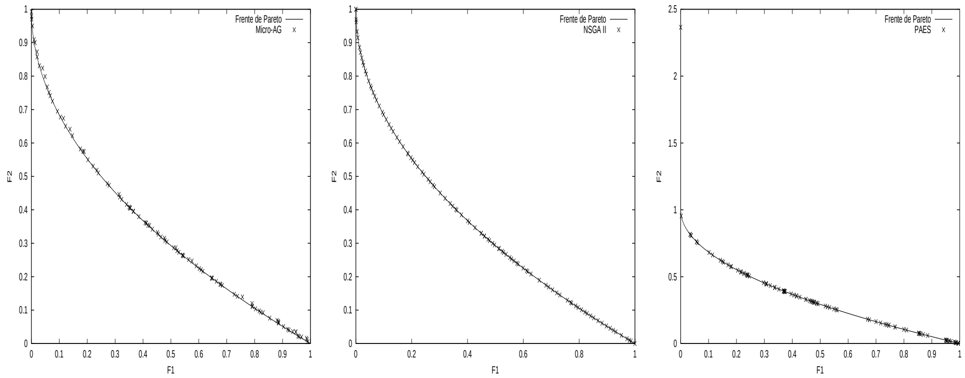


Figura 4.16: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica $M1$ (MOP3).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

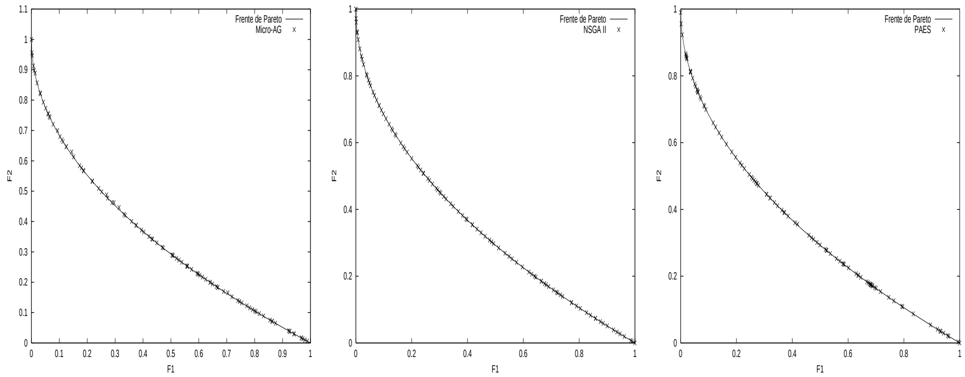


Figura 4.17: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica *O* (MOP3).

4.7 MOP 4

El cuarto problema es un problema de optimización multiobjetivo propuesto por Kursawe [42]:

$$\text{Minimize } f_1(\vec{x}) = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2+x_{i+1}^2}\right)\right) \tag{4.16}$$

$$\text{Minimize } f_2(\vec{x}) = \sum_{i=1}^{n}\left(|x_i|^{0.8}+5\sin(x_i)^3\right) \tag{4.17}$$

donde:

$$-5 \leq x_1, x_2, x_3 \leq 5 \tag{4.18}$$

Este problema tiene la característica de tener tanto el conjunto óptimo de Pareto como el frente de Pareto desconectados, cabe mencionar que la complejidad principal de este problema reside en ausencia natural de población en la región donde se encuentra el óptimo de Pareto, esta dificultad es mostrada en el espacio de la funciones objetivo en la figura 4.18.

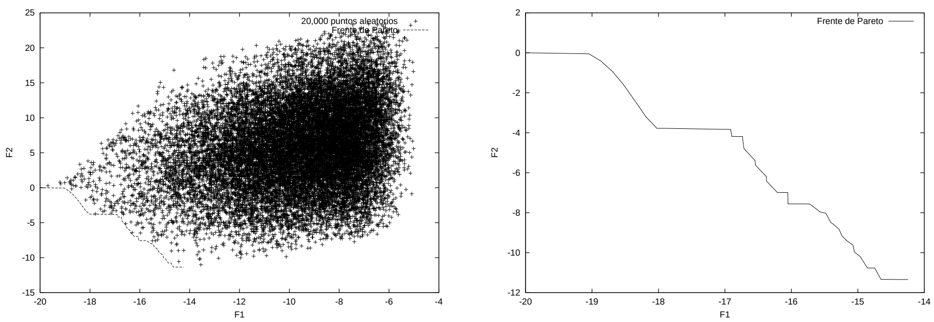


Figura 4.18: La gráfica de la izquierda muestra 20,000 soluciones generadas aleatoriamente y la derecha su frente de Pareto verdadero (MOP 4).

El número de evaluaciones de la función de aptitud que realizaron los tres algoritmos en cada una de las 20 corridas para este problema fue de 24,000. El micro-AG tuvo un tiempo promedio de ejecución en las corridas de 0.704 segundos, el NSGA II de 6.81 segundos, y el PAES de 2.195 segundos.

Los parámetros modificados por algoritmo son enumerados a continuación:

- Micro-AG:

  1. Número de iteraciones = 3,000.
  2. Porcentaje de mutación = 0.019.

- NSGA II:

1. Porcentaje de mutación = 0.333.

- PAES:

1. Porcentaje de mutación = 0.019.

4.7.1 Resultados de las métricas aplicadas

4.7.1.1 Métrica de la tasa de error

En la tabla 4.13 se pueden observar las estadísticas obtenidas de la aplicación de la métrica ER a veinte corridas de cada uno de los algoritmos. En la figura 4.19 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	2	1	12
Máximo	20	7	101
Promedio	10.7	3.3	61.35
Desviación Estándar	4.156162197	1.780005914	23.629319432
Varianza	17.273684211	3.168421053	558.344736842

Tabla 4.13: Tabla de estadísticas de ER (MOP 4).

4.7.1.2 Métrica de espacio distribuido

En la tabla 4.14 se pueden observar las estadísticas obtenidas de la aplicación de la métrica denominada espacio distribuido a veinte corridas de cada uno de los algoritmos y en la figura 4.20 se grafican las peores, medianas y mejores corridas por algoritmo.

4.7.1.3 Métrica M1

En la tabla 4.15 se pueden observar las estadísticas obtenidas de la aplicación de la métrica $M1$ a veinte corridas de cada uno de los algoritmos y en la figura 4.21 se

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.0392858	0.0392457	0.0275747
Máximo	0.0460645	0.0448214	0.0436152
Promedio	0.042321675	0.04185293	0.03715132
Desviación Estándar	0.001699098	0.001483825	0.004419295
Varianza	0.000002887	0.000002202	0.00001953

Tabla 4.14: Tabla de estadísticas de la métrica de espacio distribuido (MOP 4).

Estadística	Micro-AG	NSGA II	PAES
Mínimo	0.119881	0.130859	0.188075
Máximo	0.146472	0.153633	1.19025
Promedio	0.13460185	0.13777005	0.42445655
Desviación estándar	0.00692098961797433	0.00537119552971469	0.254083189458943
Varianza	4.79000972921084e-05	2.8849741418427e-05	0.064558267165629

Tabla 4.15: Tabla de estadísticas de M1 (MOP 4).

Estadística	Micro-AG	NSGA II	PAES
Mínimo	0.0835861	0.106183	0.342847
Máximo	0.1267	0.228852	1.30459
Promedio	0.10418712	0.14673635	0.57414095
Desviación Estándar	0.011793625	0.033148274	0.214583293
Varianza	0.00013909	0.001098808	0.046045989
Mediana	0.103298	0.13859	0.5185635

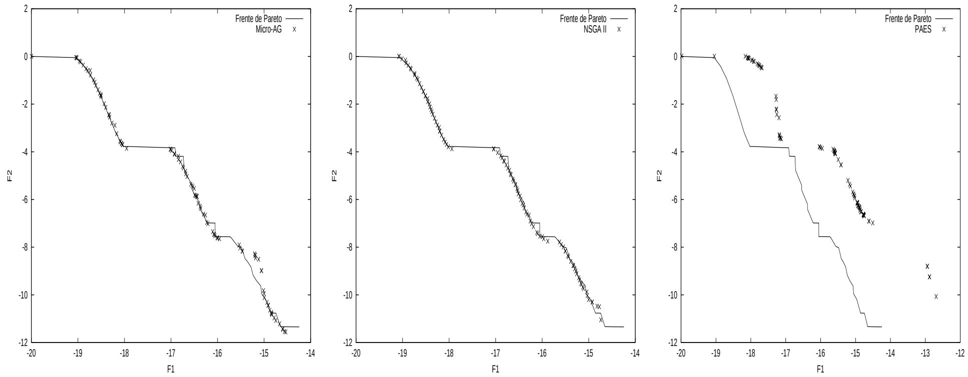
Tabla 4.16: Tabla de estadísticas de métrica O (MOP 4).

grafican las peores, medianas y mejores corridas por algoritmo.

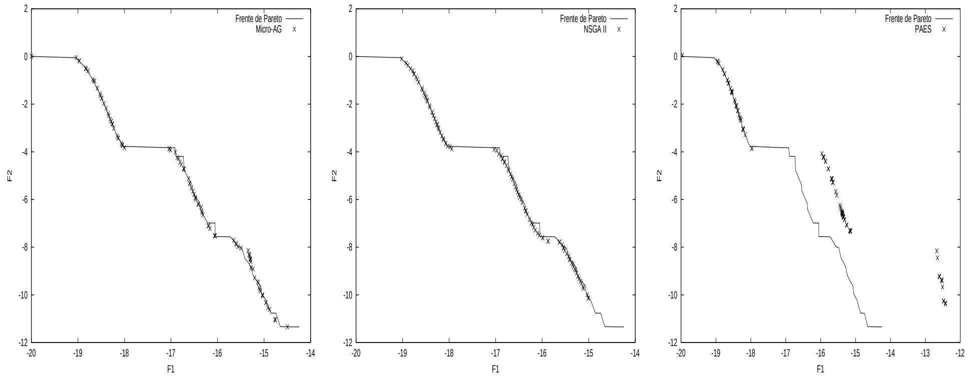
4.7.1.4 Métrica 0

En la tabla 4.16 se pueden observar las estadísticas obtenidas de la aplicación de la métrica O a veinte corridas de cada uno de los algoritmos. En la figura 4.22 se grafican las peores, medianas y mejores corridas por algoritmo.

Frente de Pareto correspondiente a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frente de Pareto correspondiente a las mejores corridas.

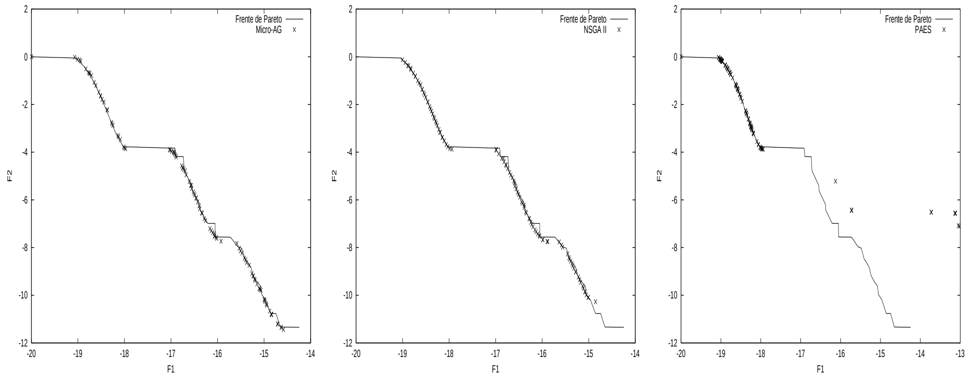
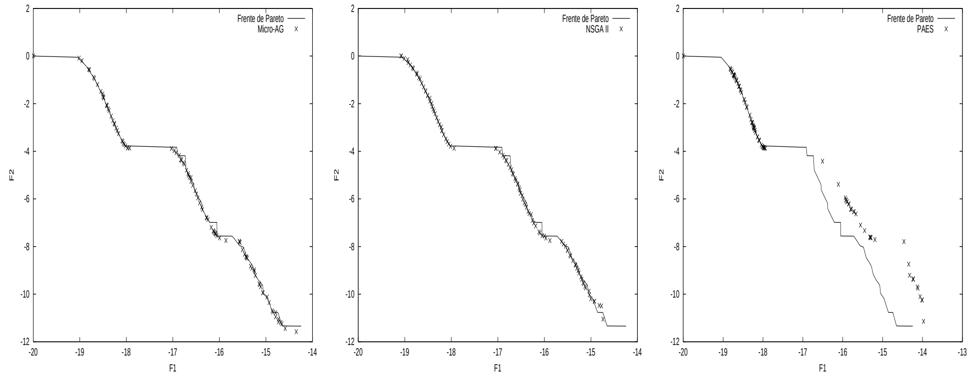
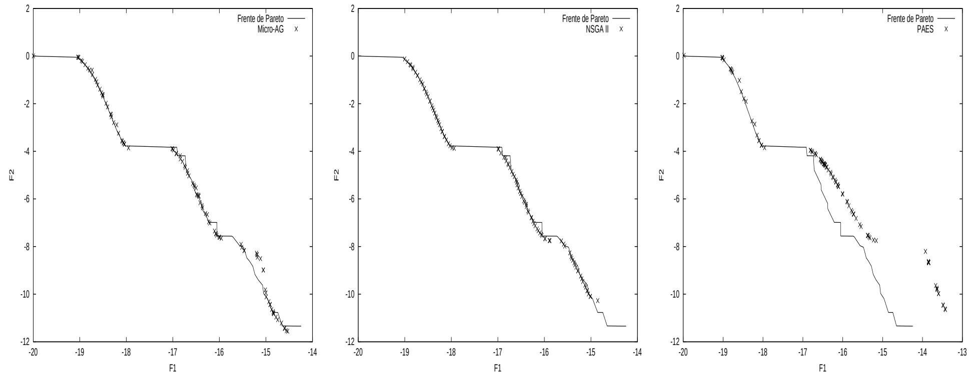


Figura 4.19: Frente de Pareto correspondiente a las peores, medianas y mejores corridas de acuerdo a la métrica ER (MOP4).

Frente de Pareto correspondiente a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frente de Pareto correspondiente a las mejores corridas.

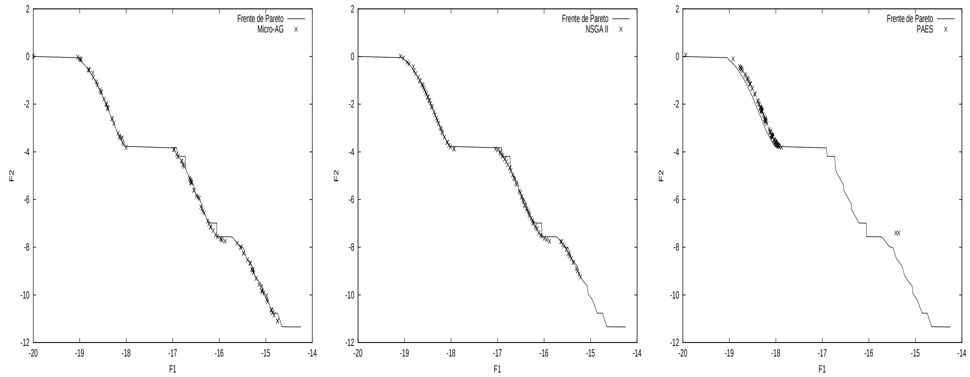
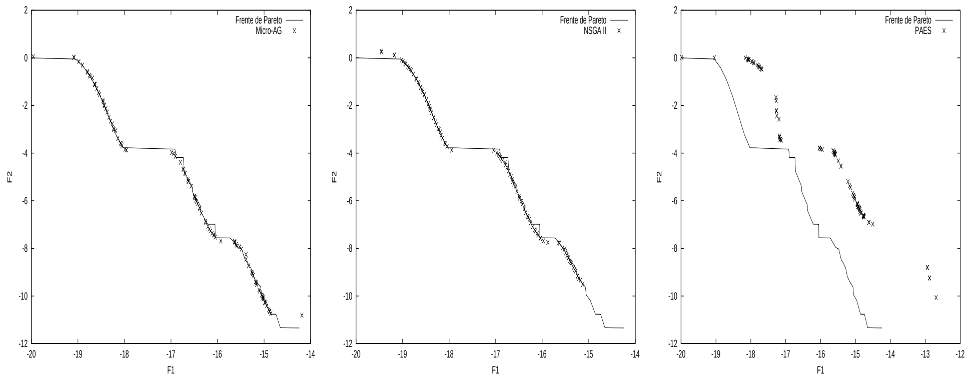
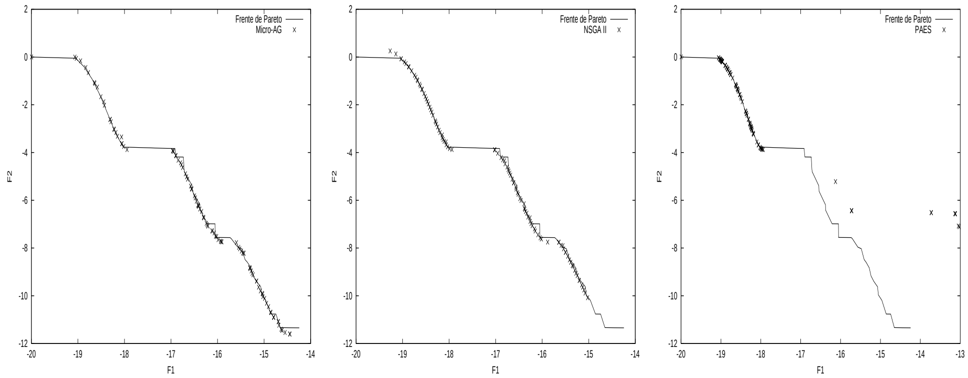


Figura 4.20: Frente de Pareto correspondiente a las mejores corridas de acuerdo a la métrica de *espacio distribuido* (MOP4).

Frente de Pareto correspondiente a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frente de Pareto correspondiente a las mejores corridas

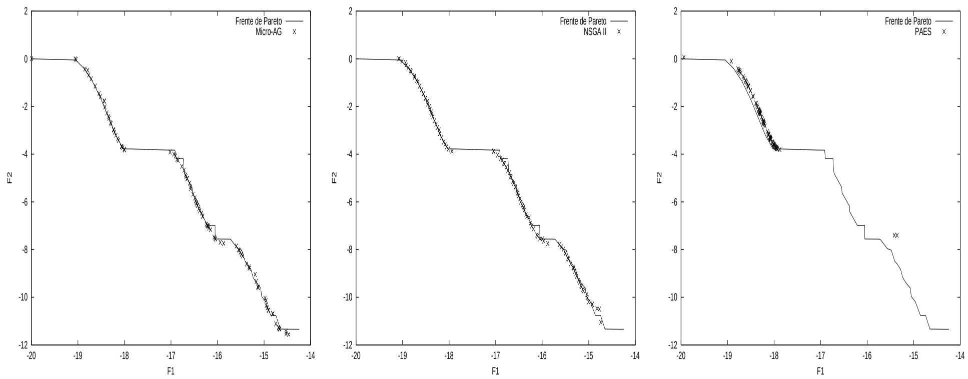
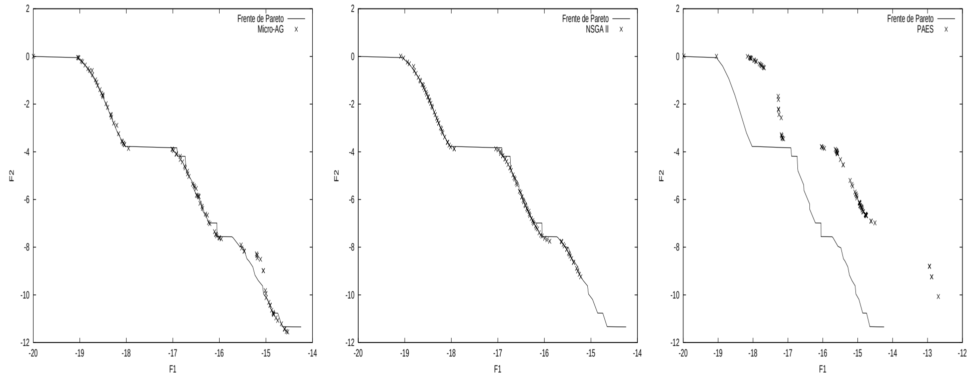
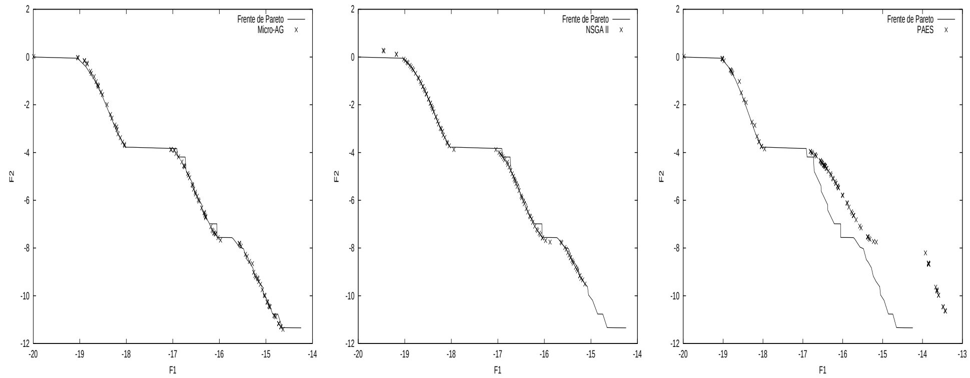


Figura 4.21: Frente de Pareto correspondiente a las mejores corridas de acuerdo a la métrica $M1$ (MOP4).

Frente de Pareto correspondiente a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frente de Pareto correspondiente a las mejores corridas.

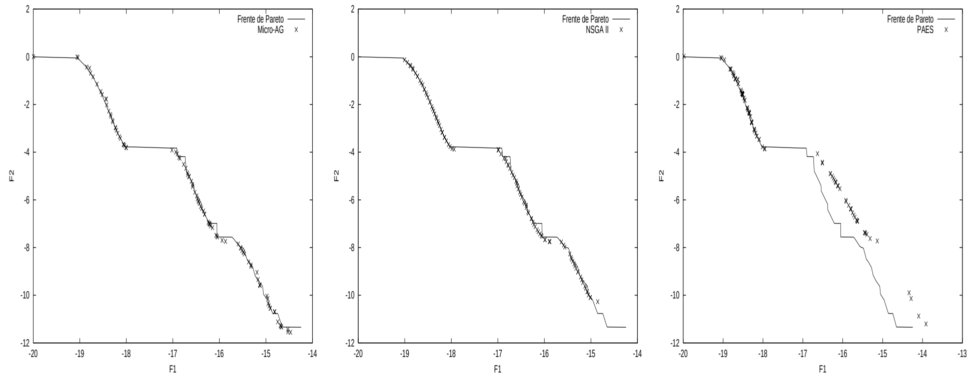


Figura 4.22: Frente de Pareto correspondiente a las mejores corridas de acuerdo a la métrica *O* (MOP4).

4.8 MOP 5

El quinto problema es un problema de optimización de dos objetivos propuesto por Kita [36]

$$\text{Maximizar } f_1(x,y) = -x^2 + y \tag{4.19}$$

$$\text{Maximizar } f_2(x,y) = \frac{1}{2}x + y + 1 \tag{4.20}$$

sujeto a las siguientes restricciones:

$$\frac{1}{6}x + y - \frac{13}{2} \leq 0 \tag{4.21}$$

$$\frac{1}{2}x + y - \frac{15}{2} \leq 0 \tag{4.22}$$

$$\frac{5}{x} + y - 30 \leq 0 \tag{4.23}$$

y $0 \leq x \leq 7.0$, $0 \leq y \leq 7.0$.

Este problema de maximización presenta un frente de Pareto cóncavo, además de diferentes restricciones que complican el espacio de búsqueda. En la figura 4.23 se muestran 20,000 soluciones aleatorias generadas para este problema así como su frente de Pareto real.

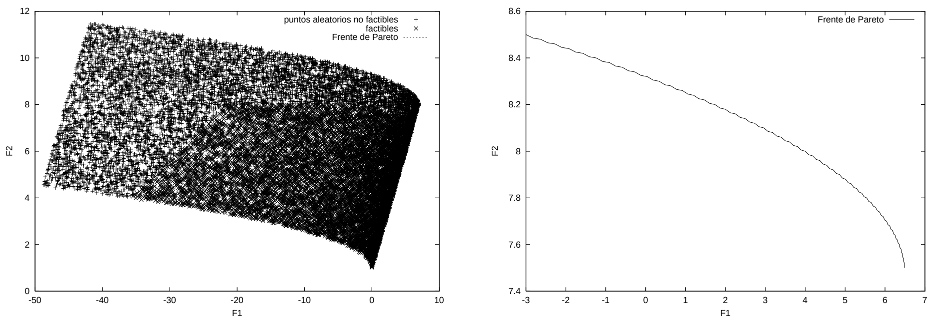


Figura 4.23: La gráfica de la izquierda muestra 20,000 soluciones generadas aleatoriamente y la derecha su frente de Pareto verdadero (MOP 5).

El número de evaluaciones de la función de aptitud que realizaron los tres algoritmos en cada una de las 20 corridas para este problema fue de 20,000. El micro-AG tuvo un tiempo promedio de ejecución en las corridas de 0.26896 segundos, el NSGA II de 6.4857 segundos, y el PAES de 58.937 segundos.

Los parámetros modificados por algoritmo son enumerados a continuación:

- Micro-AG:

    1. Número de iteraciones = 2,500.

2. Porcentaje de mutación = 0.0217.

- NSGA II:

1. Porcentaje de mutación = 0.5.

- PAES:

1. Porcentaje de mutación = 0.0217.

4.8.1 Resultados de las métricas aplicadas

4.8.1.1 Métrica de la tasa de error

En la tabla 4.17 se pueden observar las estadísticas obtenidas de la aplicación de la métrica *ER* a veinte corridas de cada uno de los algoritmos. En la figura 4.24 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	56	22	70
Máximo	87	45	101
Promedio	74.85	33.1	95.1
Desviación Estándar	7.617673357	6.471801181	7.772420879
Varianza	58.028947368	41.884210526	60.410526316

Tabla 4.17: Tabla de estadísticas de ER (MOP 5).

4.8.1.2 Métrica espacio distribuido

En la tabla 4.18 se pueden observar las estadísticas obtenidas de la aplicación de la métrica denominada espacio distribuido a veinte corridas de cada uno de los algoritmos y en la figura 4.25 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.016325	0.0146158	0.00838875
Máximo	0.0519107	0.0415921	0.0347277
Promedio	0.027347815	0.02224558	0.020602694
Desviación Estándar	0.007135252	0.007879897	0.007538593
Varianza	0.000050912	0.000062093	0.00005683

Tabla 4.18: Tabla de estadísticas de espacio distribuido (MOP 5).

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.0230817	0.013129	0.0165718
Máximo	0.962825	0.28222	2.26375
Promedio	0.26210439	0.04684924	0.399809545
Desviación estándar	0.249536997304429	0.0806184156196428	0.49631366448126
Varianza	0.0622687130237104	0.00649932893702147	0.246327253550816

Tabla 4.19: Tabla de estadísticas de M1 (MOP 5).

4.8.1.3 Métrica M1

En la tabla 4.19 se pueden observar las estadísticas obtenidas de la aplicación de la métrica $M1$ a veinte corridas de cada uno de los algoritmos y en la figura 4.26 se grafican las peores, medianas y mejores corridas por algoritmo.

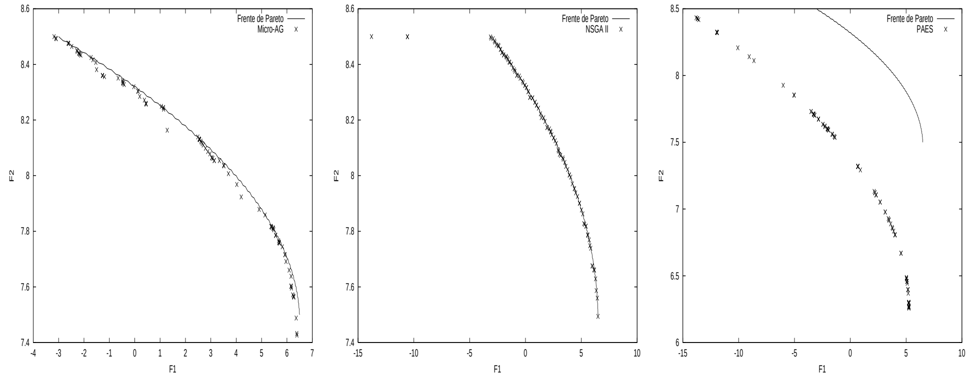
4.8.1.4 Métrica O

En la tabla 4.20 se pueden observar las estadísticas obtenidas de la aplicación de la métrica O a veinte corridas de cada uno de los algoritmos. En la figura 4.27 se grafican las peores, medianas y mejores corridas por algoritmo.

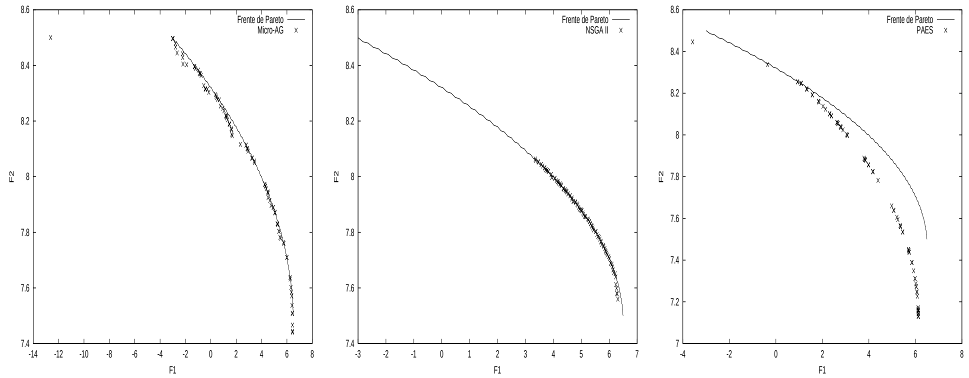
Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.0458384	0.0671157	0.19232
Máximo	0.3925	1.57741	1.59824
Promedio	0.14582332	0.646747825	0.642254
Desviación Estandar	0.089465121	0.435440834	0.421224683
Varianza	0.008004008	0.189608719	0.177430234

Tabla 4.20: Tabla de estadísticas de métrica O (MOP 5).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

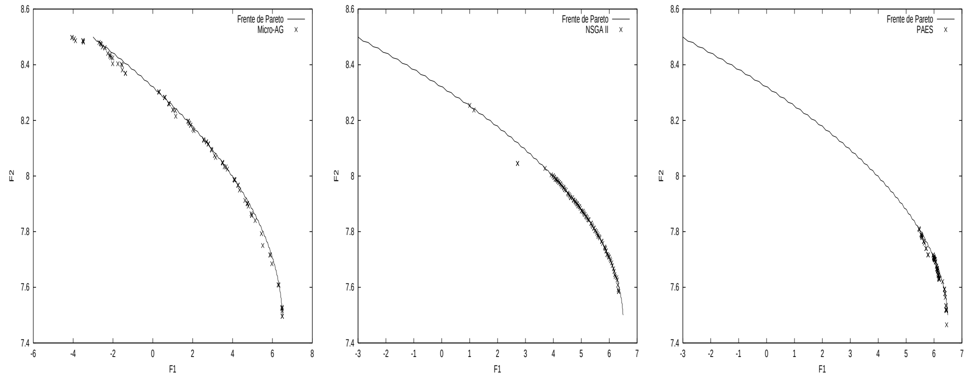
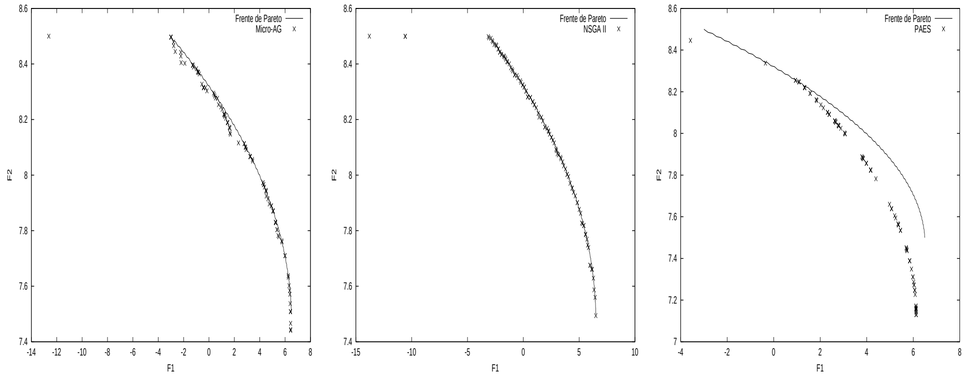
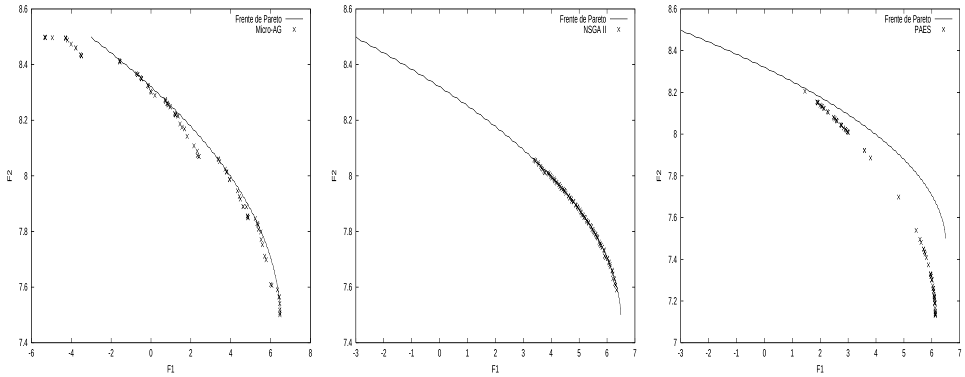


Figura 4.24: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas a la métrica *ER* (MOP5).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

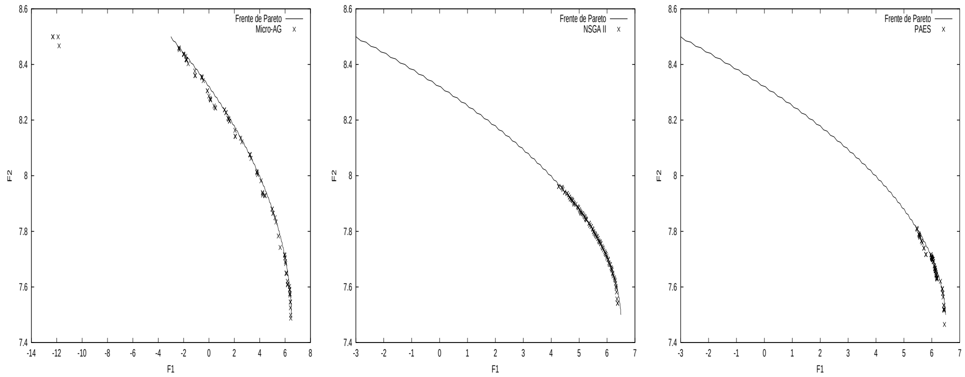


Figura 4.25: Frentes de Pareto correspondientes a las mejores corridas de acuerdo a la métrica de *espacio distribuido* (MOP5).

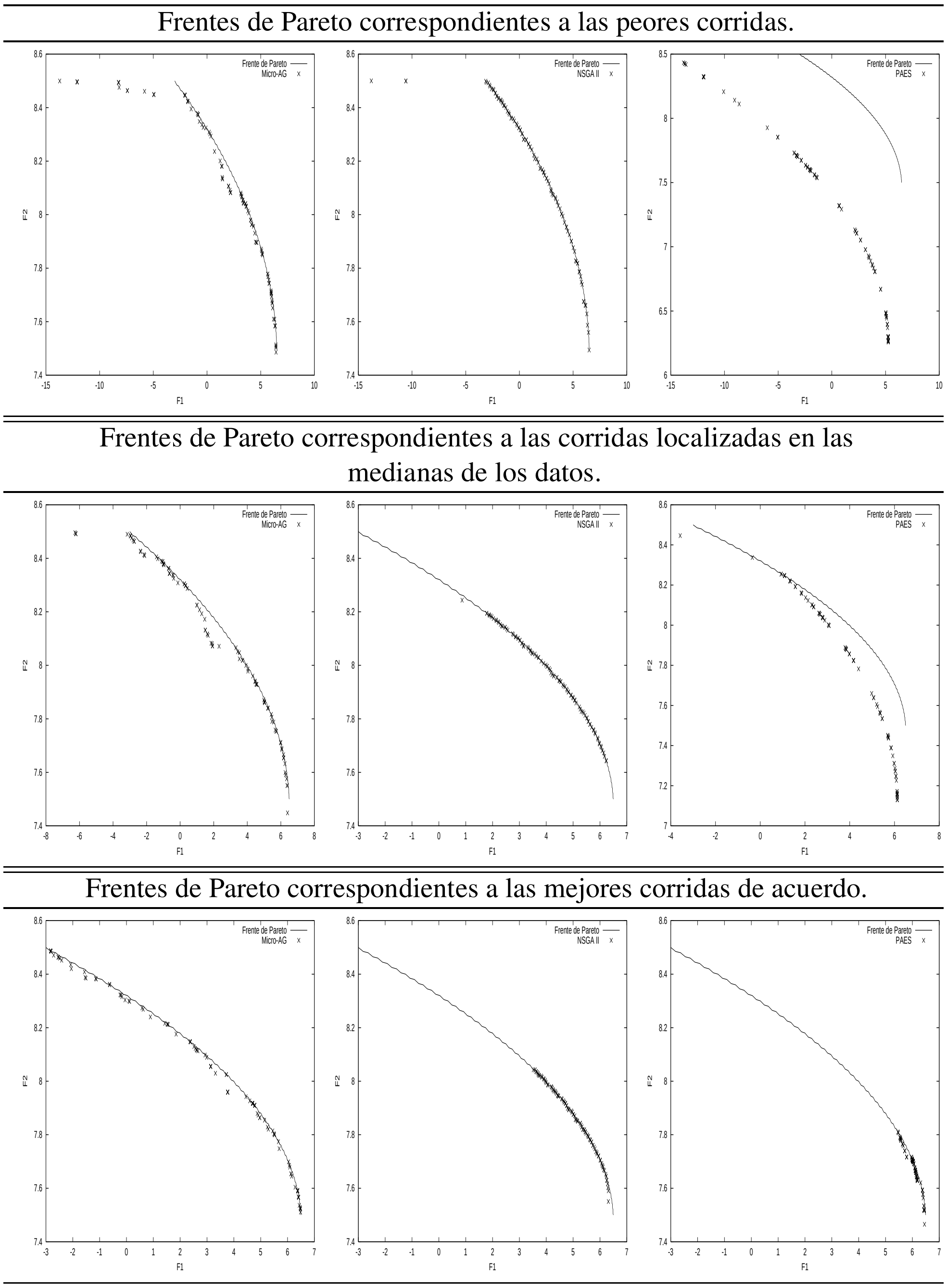
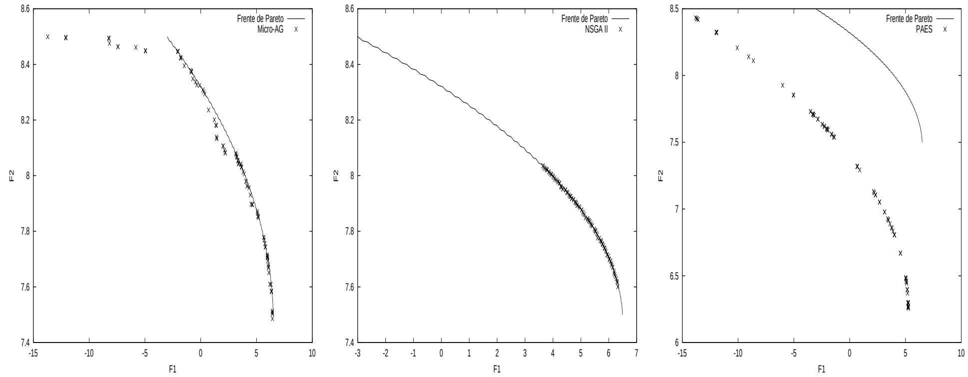
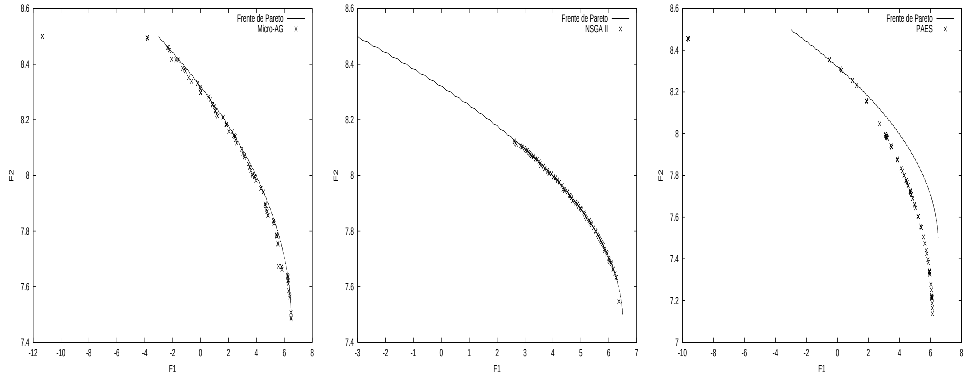


Figura 4.26: Frentes de Pareto correspondientes a las mejores corridas de acuerdo a la métrica $M1$ (MOP5).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

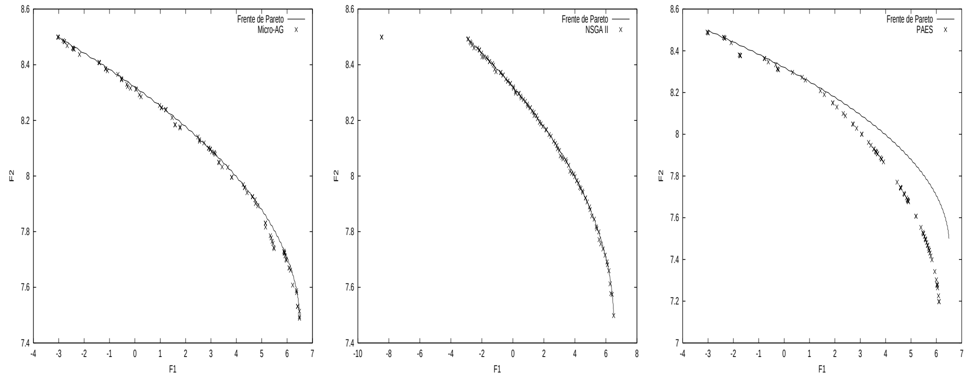


Figura 4.27: Frentes de Pareto correspondientes a las mejores corridas de acuerdo a la métrica *O* (MOP5).

4.9 MOP 6

El último ejemplo es un problema de optimización de dos objetivos propuesto por Srinivas y Deb [62].

$$\text{Minimizar } f_1(x,y) = (x-2)^2 + (y-1)^2 + 2 \tag{4.24}$$

$$\text{Minimizar } f_2(x,y) = 9x - (y-1)^2 \tag{4.25}$$

Sujeto a las siguientes restricciones:

$$x^2 + y^2 - 225 \leq 0 \tag{4.26}$$

$$x - 3y + 10 \leq 0 \tag{4.27}$$

y $-20 \leq x, y \leq 20.0$.

El presente problema presenta un conjunto óptimo de Pareto discontinuo; en la figura 4.28 se presenta una ejemplificación de lo restringido del problema mediante 20,000 soluciones generadas aleatoriamente aunado a su frente de Pareto real.

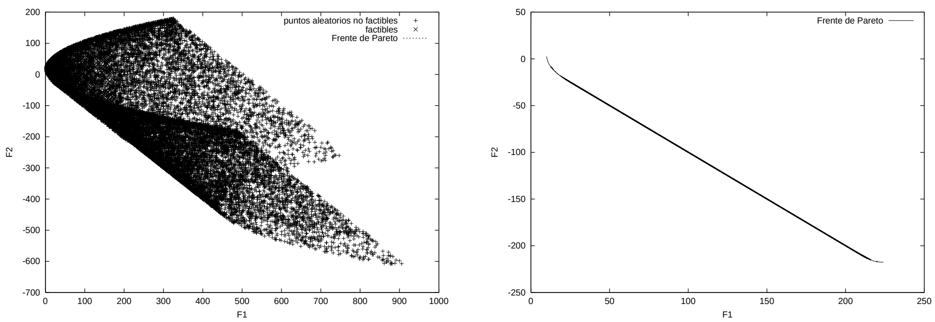


Figura 4.28: La gráfica de la izquierda muestra 20,000 soluciones generadas aleatoriamente y la derecha su frente de Pareto verdadero (MOP 6).

El número de evaluaciones de la función de aptitud que realizaron los tres algoritmos en cada una de las 20 corridas para este problema fue de 12,000. El micro-AG tuvo un tiempo promedio de ejecución en las corridas de 3.4679 segundos, el NSGA II de 4.038 segundos, y el PAES de 56.6706 segundos.

Los parámetros modificados por algoritmo son enumerados a continuación:

- Micro-AG:
  1. Número de iteraciones = 1,500.
  2. Porcentaje de mutación = 0.0192.
- NSGA II:
  1. Porcentaje de mutación = 0.5.

- PAES:
    1. Porcentaje de mutación = 0.0192.

4.9.1 Resultados de las métricas aplicadas

4.9.1.1 Métrica de la tasa de error (ER)

En la tabla 4.21 se pueden observar las estadísticas obtenidas de la aplicación de la métrica ER a veinte corridas de cada uno de los algoritmos. En la figura 4.29 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	33	30	8
Máximo	77	50	101
Promedio	55	42.5	54.3
Desviación Estándar	10.843382653	5.501196042	34.788231525
Varianza	117.578947368	30.263157895	1210.221052632

Tabla 4.21: Tabla de estadísticas de ER (MOP 6).

4.9.1.2 Métrica espacio distribuido

En la tabla 4.22 se pueden observar las estadísticas obtenidas de la aplicación de la métrica denominada espacio distribuido a veinte corridas de cada uno de los algoritmos y en la figura 4.30 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	**Micro-AG**	**NSGA II**	**PAES**
Mínimo	0.187552	0.236813	0.0830411
Máximo	0.227054	0.258613	0.209255
Promedio	0.2087396	0.247253	0.146405255
Desviación Estándar	0.010295309	0.005937637	0.031446216
Varianza	0.000105993	0.000035256	0.000988864

Tabla 4.22: Tabla de estadísticas de la métrica de espacio distribuido (MOP 6).

4.9.1.3 Métrica M1

En la tabla 4.23 se pueden observar las estadísticas obtenidas de la aplicación de la métrica $M1$ a veinte corridas de cada uno de los algoritmos y en la figura 4.31 se grafican las peores, medianas y mejores corridas por algoritmo.

Estadística	Micro-AG	NSGA II	PAES
Mínimo	0.210846	0.238071	0.125562
Máximo	0.798591	0.387376	88.6098
Promedio	0.4046362	0.2951232	5.6864776
Desviación estándar	0.146388379945303	0.0389762231226328	19.5991645653948
Varianza	0.0214295577830105	0.00151914596890526	384.127251661429

Tabla 4.23: Tabla de estadísticas de M1 (MOP 6).

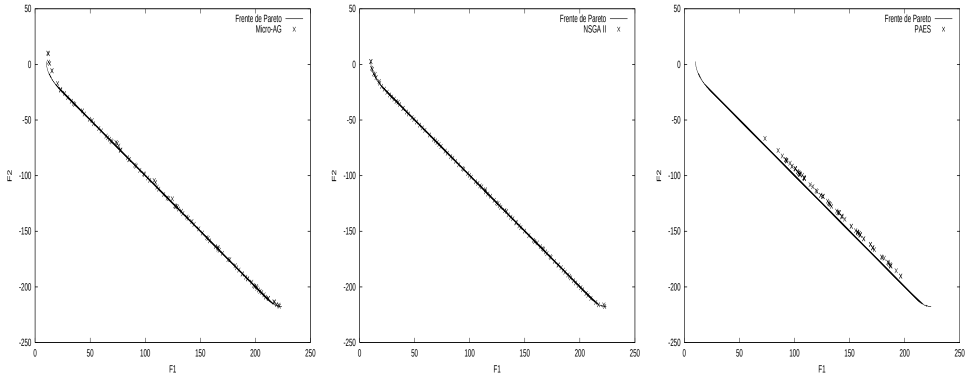
4.9.1.4 Métrica 0

En la tabla 4.24 se pueden observar las estadísticas obtenidas de la aplicación de la métrica O a veinte corridas de cada uno de los algoritmos. En la figura 4.32 se grafican las peores, medianas y mejores corridas por algoritmo.

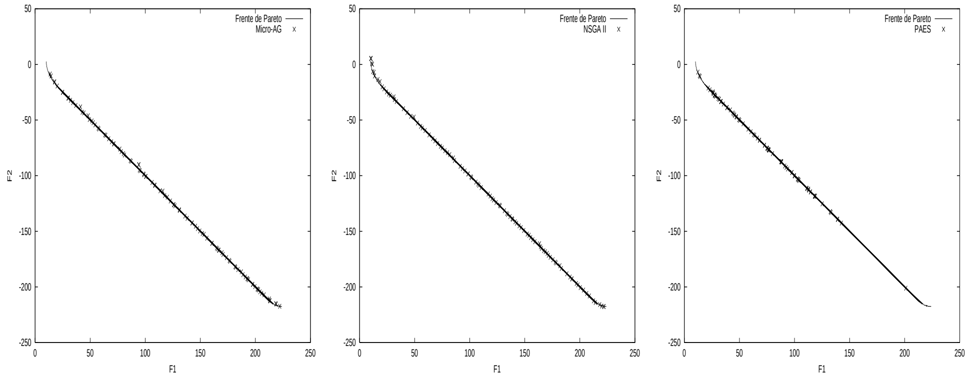
Estadística	Micro-AG	NSGA II	PAES
Mínimo	0.959901	0.682054	1.72038
Máximo	1.37197	0.883083	109.117
Promedio	1.132273	0.79185975	10.79305
Desviación Estándar	0.130674067	0.056466367	23.469314765
Varianza	0.017075712	0.003188451	550.808735531

Tabla 4.24: Tabla de estadísticas de métrica O (MOP 6).

Frentes de Pareto correspondientes s a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos



Frentes de Pareto correspondientes a las mejores corridas

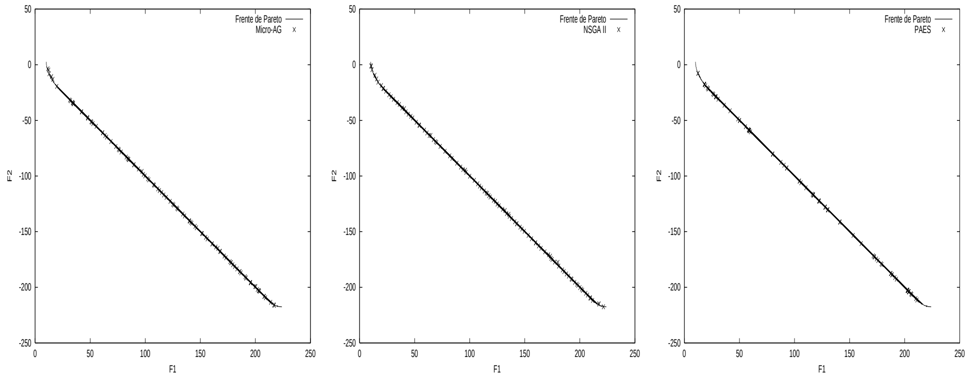
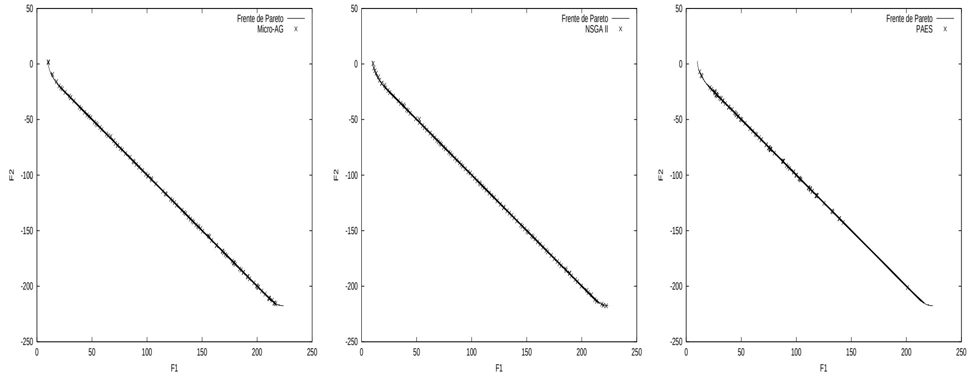
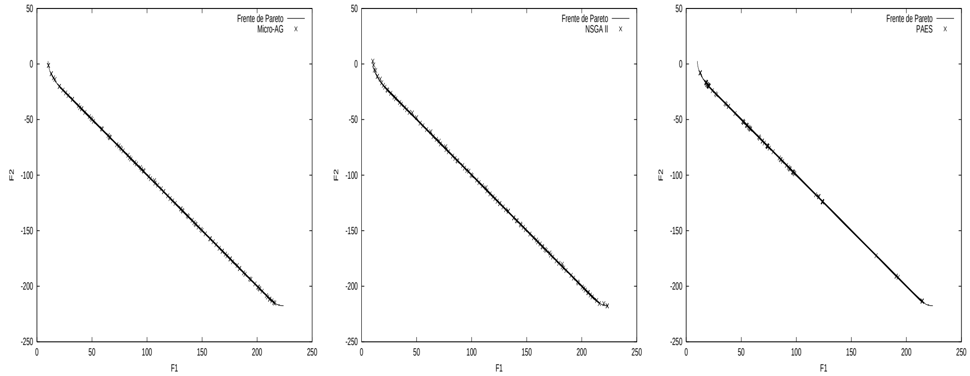


Figura 4.29: Gráficas de las peores, medianas y mejores corridas de acuerdo a la métrica ER (MOP6).

Frentes de Pareto correspondientes a las peores corridas.



Frentes de Pareto correspondientes a las corridas localizadas en las medianas de los datos.



Frentes de Pareto correspondientes a las mejores corridas.

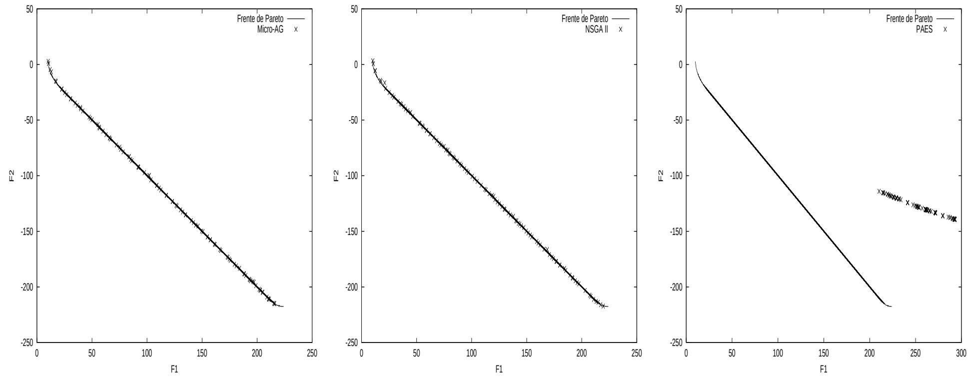


Figura 4.30: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas a la métrica de *espacio distribuido* (MOP6).

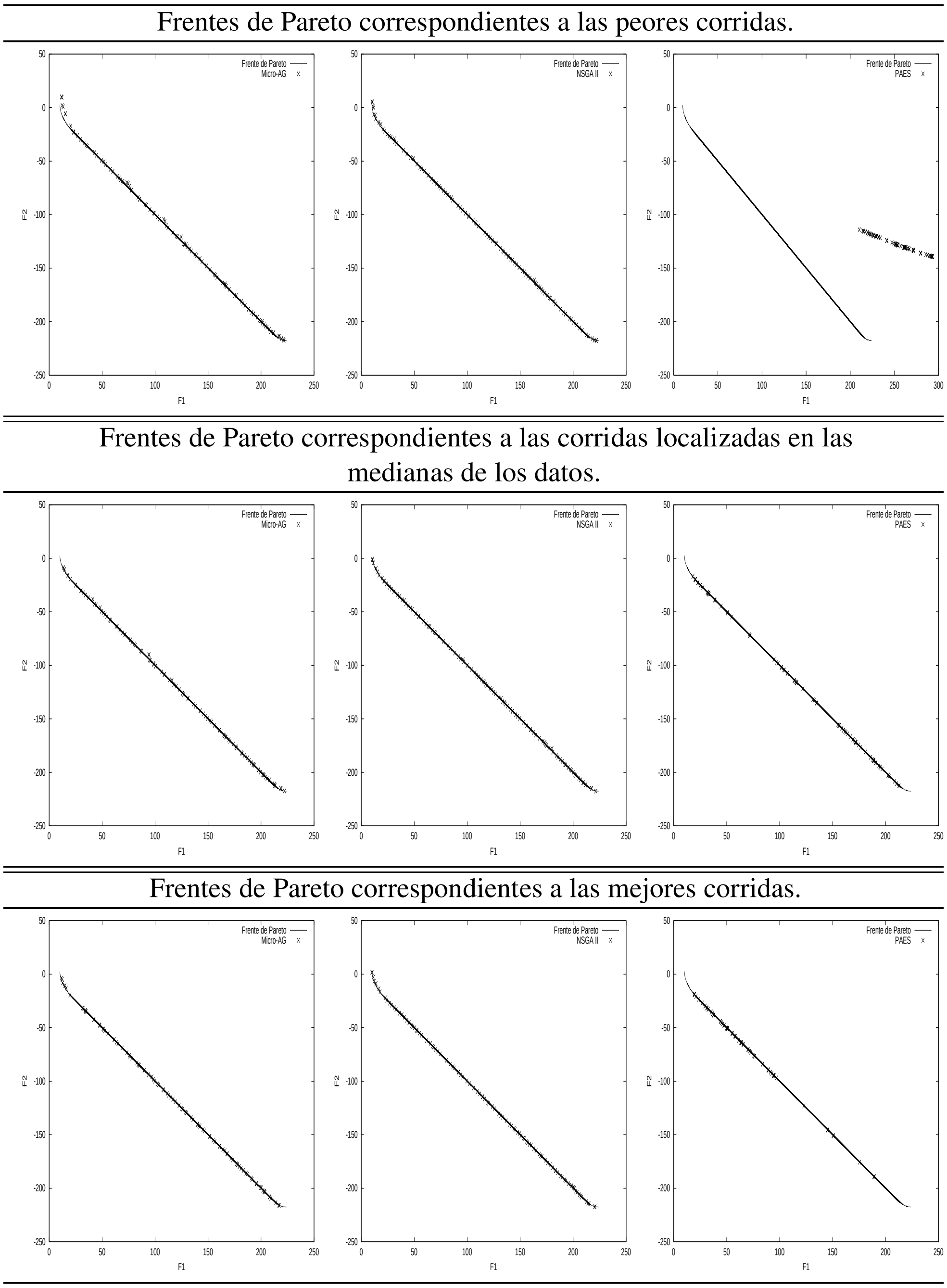


Figura 4.31: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica $M1$ (MOP6).

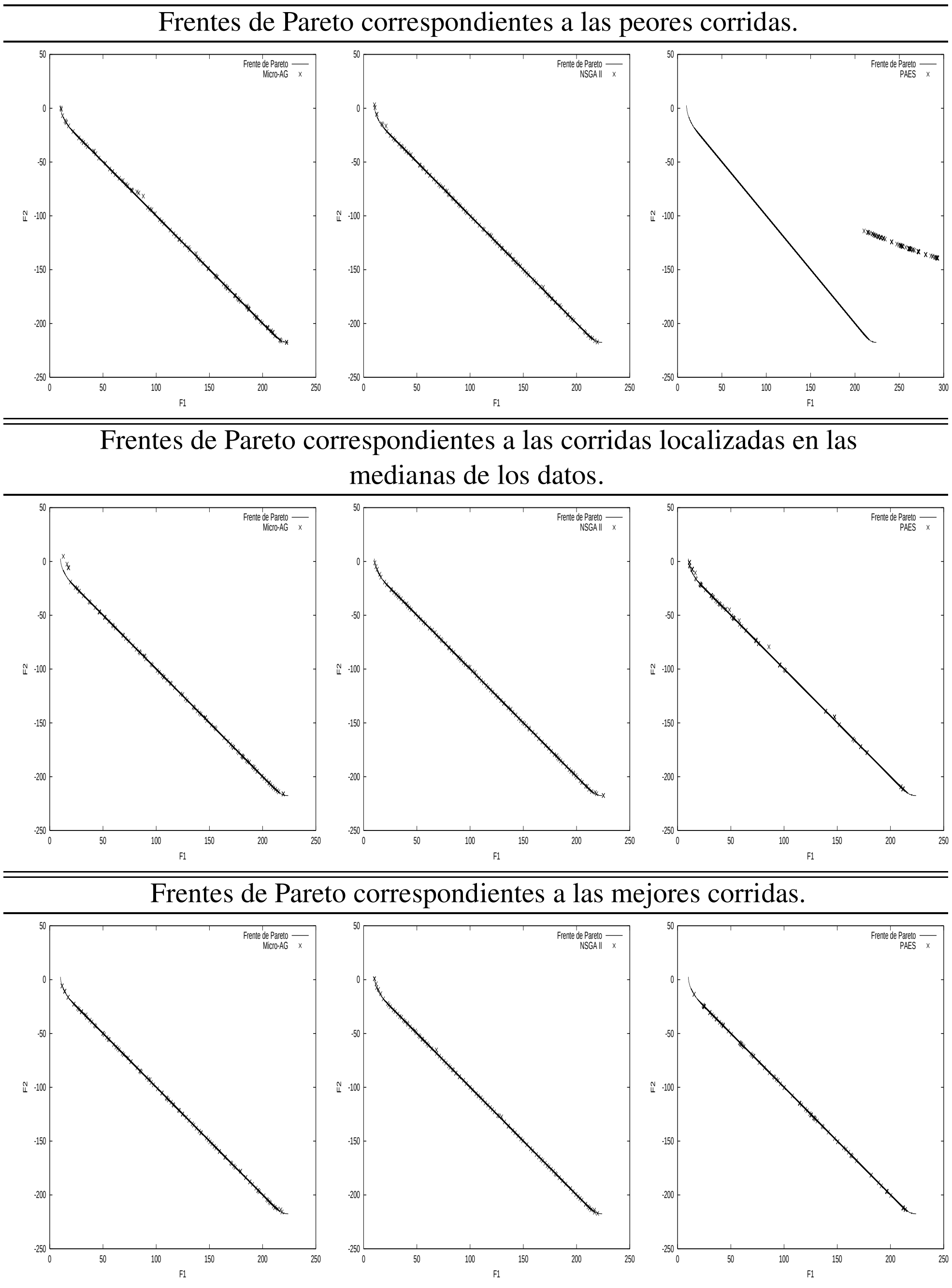


Figura 4.32: Frentes de Pareto correspondientes a las peores, medianas y mejores corridas de acuerdo a la métrica *O* (MOP6).

4.10 Análisis de resultados

Irónicamente, la medición cuantitativa que permite comparar entre diferentes técnicas evolutivas mutiobjetivo es a su vez, un problema de optimización multiobjetivo, en donde se trata de encontrar que técnica (dentro de un conjunto de ellas) se aproxima, abarca y se distribuye mejor con respecto al verdadero frente de Pareto. Indiscutiblemente, ninguna de las métricas utilizadas, puede dar por sí sola, una descripción cabal del comportamiento de las técnicas involucradas y además brindar las bases necesarias para poder hacer una comparación justa. Sin embargo, al conjuntar las cuatro métricas es posible realizar una mejor comparación, mediante el análisis cuantitativo de sus desempeños en cada función de prueba (aunque no pueda ser elegida una técnica vencedora en términos generales[1]).

Una descripción de ciertas debilidades y fortalezas de las tres técnicas, se proporcionan a continuación:

NSGA II es un algoritmo eficiente, que provee soluciones elegantes. Esto se debe, en gran medida a su condición elitista, la cual permite explotar espacios de búsqueda promisorios. Sin embargo, no tiene una capacidad exploratoria buena. Esto es claramente visible en la función de prueba 4, donde en ninguna de las 20 corridas pudo encontrar el punto $[-20,0]$. Esta debilidad también se puede observar en la función de prueba 5, donde jamás logró extender soluciones a lo largo del frente de Pareto, alojando la gran mayoría en una pequeña porción de éste. Con esto expuesto se piensa que se puede mejorar la capacidad exploratoria del NSGA II e incrementar su desempeño. Finalmente, aunque el desempeño de este algoritmo fue muy bueno en las funciones probadas en esta tesis, cabe mencionar que también presentó la mayor cantidad tiempo de cómputo por corrida, lo cual es, una clara desventaja.

PAES es un algoritmo simple, rápido, y eficiente. Existen funciones de prueba en las que PAES se desempeñó muy bien. Sin embargo, en muchas otras su desempeño no fue aceptable. Esto se debe en gran medida a la carencia de una adecuada selección de los individuos para evolucionar (como los individuos que se encuentran en los extremos y aquellos individuos que pertenecen a regiones poco pobladas del frente de Pareto). Esta deficiencia en la selección también se refleja en su pobre desempeño en funciones con frente y conjunto de óptimos de Pareto discontinuo, ya que puede estar saltando de una región a otra sin jamás poder ninguna explotar adecuadamente. Este comportamiento es evidente en la funciones de prueba 3 y 4 y en la función de prueba (con restricciones) 6 (en esta última, también puede influir la metodología usada para el manejo de restricciones). Finalmente se estima que

[1]Esto es de esperarse, pues es una consecuencia del "No Free Lunch Theorem" [70]. Sin embargo, en este caso en particular nos referimos a que, dado un conjunto de funciones de prueba y ciertos algoritmos evolutivos multiobjetivo, suele resultar difícil determinar una clara superioridad de cualquiera de dichos algoritmos, aún usando métricas como las adoptadas en esta tesis.

se puede mejorar el desempeño de la malla adaptativa, ya que ésta es la causa de la gran cantidad de tiempo consumida, por ejemplo, en las corridas de la función 5.

Micro-AG es un algoritmo concebido para explotar las ventajas del elitismo. Sin embargo, también se creó con una robusta capacidad exploratoria. Se podría decir que cuenta con la capacidad exploratoria de PAES y una aproximación (aunque un tanto burda) de la capacidad explotatoria del NSGA II. Lo anterior puede ser confirmado en las funciones 4 y 5 donde el micro-AG pudo localizar puntos que el NSGA II no pudo y tuvo una mejor dispersión en el frente de Pareto real. Sin embargo, el micro-AG presenta serias debilidades al presentársele problemas con restricciones, por lo que se intuye que con un método más apropiado para manejar restricciones, el algoritmo tendría un desempeño más robusto. Otra debilidad, es el faltante de un poder de explotación eficiente (como el mostrado en el NSGA II). Otro factor en contra del micro-AG es la cantidad excesiva de parámetros. Finalmente, otra característica alentadora del micro-AG, fue el bajo consumo computacional que presentó constantemente a lo largo de las diferentes corridas, manteniendo el tiempo más bajo de ejecución de los tres algoritmos.

Para poder brindar el análisis comparativo deseado, en la figura 4.33 se puede observar la graficación del mejor, peor y desempeño promedio de la aplicación de las diferentes métricas a las corridas de los tres algoritmos. Con base en estos resultados a se realiza a continuación un análisis por función:

Función de prueba 1 El micro-AG fue el claro vencedor de los algoritmos comparados, al haber tenido el mejor desempeño en las cuatro métricas.

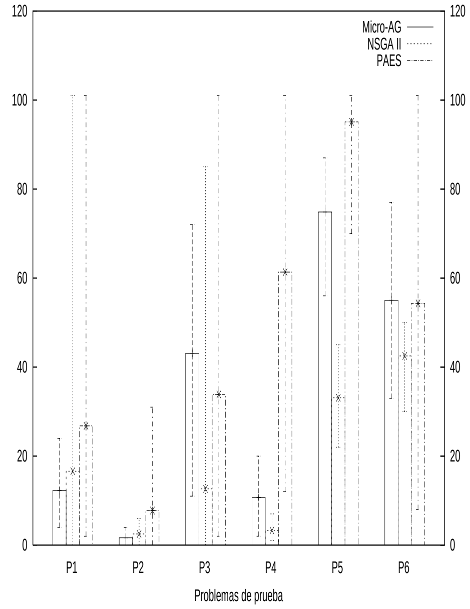
Función de prueba 2 En esta función una vez más el micro-AG obtuvo un mejor desempeño en más métricas que las técnicas contendientes, habiendo ganado en la métrica ER y $M1$, quedando ubicado en un segundo sitio tanto en la métrica de *espacio distribuido* detrás de PAES. También en la métrica O se ubicó detrás del NSGA II.

Función de prueba 3 Nuevamente en esta función el micro-AG vence a los otros algoritmos. En esta ocasión tuvo un mejor desempeño promedio en tres métricas. Cabe mencionar que en todas las corridas el micro-AG tuvo una mejor corrida que resultó mejor que las de los algoritmos restantes lo que evidencia su robustez.

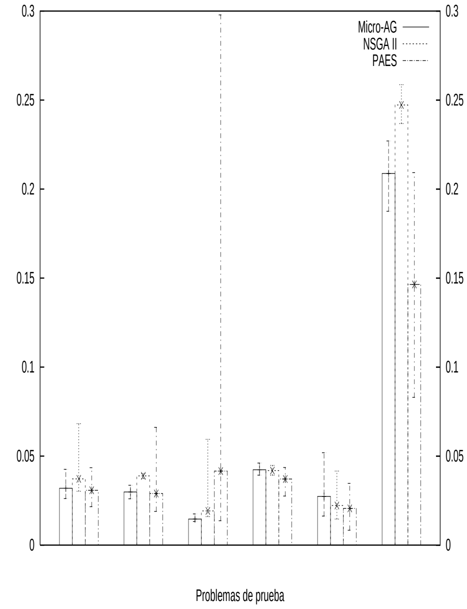
Función de prueba 4 En esta función, al igual que en la función de prueba 2, el micro-AG tiene un mejor desempeño promedio en dos métricas ($M1$ y O), y le cede el primer puesto al NSGA II en la métrica ER y a PAES en la métrica de *espacio distribuido*.

Función de prueba 5 NSGA II tuvo un mejor desempeño promedio en dos de las cuatro métricas en esta ocasión (ER y $M1$); mientras que PAES fue el mejor en la métrica de *espacio distribuido* y el micro-AG en la métrica O. En este problema el micro-AG mantuvo el segundo puesto en la métrica $M1$. Este problema es interesante, ya que la métrica O muestra una clara ventaja en la forma en como el micro-AG cubre el verdadero frente de Pareto en contraste a las otras dos técnicas (esto puede ser visualizado en la figura 4.27).

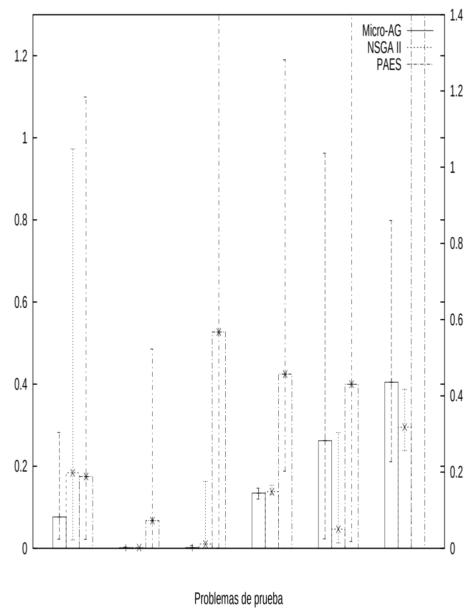
Función de prueba 6 En esta ocasión, indiscutiblemente el vencedor es el NSGA II, ya que logra un mejor desempeño promedio en tres de las cuatro métricas, únicamente vencido en la métrica de *espacio distribuido* por PAES. En esta función de prueba el micro-AG mantuvo el segundo lugar en las métricas de *espacio distribuido*, $M1$ y O.



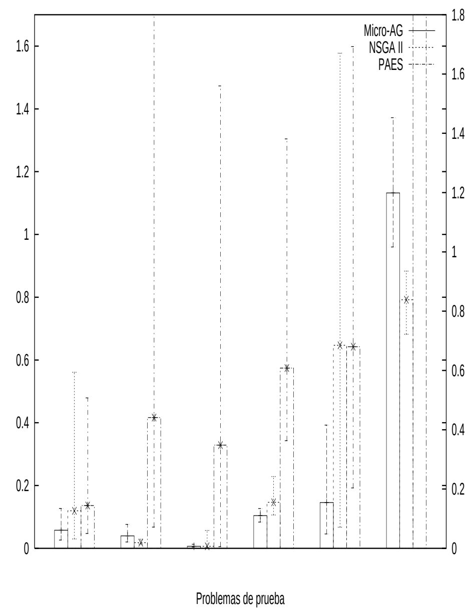
(a) Graficación del valor promedio de la aplicación de ER a las 20 corridas.



(b) Gráfica generada del promedio la aplicación de la métrica de *espacio distribuido*a todos los problemas.



(c) Gráfica generada del promedio de la aplicación de $M1$ a todos los problemas.



(d) Gráfica generada del promedio la aplicación de O a todos los problemas.

Figura 4.33: Gráficas de la abstracción de resultados de todas las métricas.

Conclusiones

El objetivo principal de esta tesis fue el de proponer una nueva técnica de optimización multiobjetivo, basada en las nuevas tendencias del área. Como resultado, se propuso usar un algoritmo genético con una población muy pequeña (solamente cuatro individuos) con un proceso de reinicialización, un archivo histórico y una malla adaptativa para manejar la diversidad en las soluciones de dicho archivo.

La técnica propuesta fue comparada contra NSGA II y PAES en diferentes funciones de prueba. En las funciones sin restricciones usadas, esta técnica fue capaz de converger más rápido (en términos de tiempo de CPU) hacia el verdadero frente de Pareto que los otros dos algoritmos. También, se puede decir que tuvo un desempeño mejor y más estable, de acuerdo a las métricas utilizadas en estas funciones. En las funciones con restricciones, sin embargo, su desempeño no fue tan bueno como el NSGA II, aunque sí mejor que PAES. De lo anterior, puede inferirse que la técnica para manejar restricciones no es la más adecuada. Otros resultados encontrados en la realización de esta tesis se presentan a continuación:

- El elitismo (o sea, usar una población secundaria para retener a las soluciones no dominadas producidas a lo largo del proceso evolutivo) es una característica imprescindible para la creación de un algoritmo eficiente. Promueve la velocidad en la convergencia al impedir perdidas en las mejores soluciones previamente encontradas y al proporcionar un camino promisorio de búsqueda.

- Una selección cuidadosa de los individuos a recombinar, proporciona una capacidad de explotación excelente, y conjugándolo con un elitismo apropiado, se pueden poblar más rápida y eficientemente los espacios promisorios, dando con ello una mayor probabilidad de encontrar nuevas soluciones no dominadas.

- Es necesario que convivan activamente, una capacidad explorativa robusta y la explotación eficiente de una región, de tal manera que se complementen; la primera, encontrando espacios de búsqueda atractivos y la segunda, poblándolos rápidamente.

- La malla adaptativa es un mecanismo que brinda simplicidad y eficiencia al manejo de diversidad en la población usando un procedimiento geográfico aplicado en el espacio de las funciones objetivo. Además, puede ser un vehículo para manejar adecuadamente la incorporación de preferencias.

- Aún utilizando diferentes métricas para medir el desempeño de algoritmos para optimización multiobjetivo, es difícil lograr unánimemente a un vencedor. De esto se intuye que mediante la creación de una técnica basada en una metodología multiobjetivo, que involucre a todas las técnicas a comparar, es posible solucionar este problema de medición de desempeño.

Trabajo futuro

Existen diversas extensiones posibles que pueden realizarse a la propuesta presentada en esta tesis. Las que se consideran como más importantes se describen a continuación:

- Dado que el micro-AG presenta su peor comportamiento en funciones de prueba con restricciones, es necesario estudiar otros esquemas posibles de manejo de restricciones que puedan acoplarse al micro-AG, a fin de subsanar los problemas del usado actualmente.
- Mejorar la capacidad del algoritmo para explotar determinada región promisoria del espacio de búsqueda (la principal ventaja del NSGA II). Esto se puede llevar a cabo mejorando la manera en que es llevada a cabo la selección de individuos en la memoria de población.
- Un cuidadoso estudio de sensibilidad de los parámetros del algoritmo es necesario para proveer mas guías generales de los valores que pueden ser establecidos por omisión en cada uno de los parámetros.
- Sería interesante usar ésta técnica como un principio de desarrollo de un modelo de incorporación de preferencias del tomador de decisiones.
- Evolucionar el micro-AG en un algoritmo auto-adaptativo, para disminuir la cantidad de parámetros y así evitar desempeños pobres del algoritmo por la elección inadecuada de los valores de los parámetros.

Bibliografía

[1] W. W. Cooper A Charnes and R. O. Ferguson. Optimal estimation of executive compensation by linear programming. *Management Science*, 1(2):138–151, 1955.

[2] a. Kaszkiewicz and R. Slowinski. The light beam serach over a non-dominated surface of a multiple-objective programming problem. In U. P. Wen G. H. Tzeng, H. F. Wand and P. L. Yu, editors, *Multiple Criteria Decision Maing- Prceedings of the Tenth International Conference: Expand and Enrich the Domains of Thinking and Application*, pages 87–99, New York, 1994. Springer-Verlag.

[3] J. S. Dyer A. M. Geoffrion and A. Feinberg. An interactive approach for multi-criterion optimization, with an application to the operation of an academic deparment. *Management Science*, 19(4):357–368, 1972.

[4] Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, dpunkt.verlag, January 1998.

[5] A. D. Belegundu and P. L. N. Murthy. A New Genetic Algorithm for Multiobjective Optimization. Technical Report AIAA-96-4180-CP, AIAA, Washington, D.C., 1996.

[6] P. J. Bentley and J. P. Wakefield. Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. In P. K. Chawdhry, R. Roy, and R. K. Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, Part 5, pages 231–240, London, June 1997. Springer Verlag London Limited. (Presented at the 2nd On-line World Conference on Soft Computing in Design and Manufacturing (WSC2)).

[7] Peter Bienert. Aufbau einer optimierungsautomatik für drei parameter. Master's thesis, Universidad Técnica de Berlin, 1967.

[8] A. Brindle. *Genetic Algorithms for Function Optimization*. PhD thesis, Edmonton, Alberta, Canada, 1981.

[9] J. T. Buchanan. A naive approach for solving mcdm problems: The guess method. *Methods, Computers & Operation Research*, 48(2):202–206, 1997.

[10] Bill P. Buckles and Fred E. Petry. *Genetic Algorithms*. IEEE Computer Society Press, 1992.

[11] V. Chankong and Y.Y. Haimes. Multiobjective decision making theory and methodology. *Elsevier Science Publishing Co.*, 1983.

[12] Carlos A. Coello Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, August 1999.

[13] Carlos A. Coello Coello. *Notas de clase*, 2 edition, enero–mayo 2000.

[14] N. L. Cramer. A representation for the adaptive generation of simple. In J.J. Grefenstette, editor, *Proceedings of an International Conference on Genetic*, pages 183–187, Pittsburgh, PA., 1985. Carnegie-Mellon University.

[15] Charles Darwin. *El origen de las especies*. Editorial Porrua, 1956.

[16] Lawrence Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[17] Kalyanmoy Deb. Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, 7(3):205–230, Fall 1999.

[18] Kalyanmoy Deb, Samir Agrawal, Amrit Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[19] Kalyanmoy Deb and David E. Goldberg. An investigation of niche and species formation in genetic function optimization. In David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, San Mateo, California, June 1989. George Manson University, Morgan Kaufmann.

[20] G. Dozier, J. Bowen, and D. Bahler. Solving small and large scale constraint satisfaction problems using a heuristic-based microgenetic algorithm. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 306–311, 1994.

[21] P. C. Fishburn. Lexicographic orders, utilities and decision rueles: A survey. *Management Science*, 20(11):1442–1471, 1974.

[22] Lawrence J. Fogel. *On the organization of intellect*. PhD thesis, University of California, Los Angeles, California, 1964.

[23] Lawrence J. Fogel. *Artificial Intelligence through Simulated Evolution. Fourty years of Evolutionary Programming*. John Wiley & Sons, New York, 1999.

[24] Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[25] Carlos M. Fonseca and Peter J. Fleming. An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1):1–16, Spring 1995.

[26] D. E. Goldberg and J. Richardson. Genetic algorithm with sharing for multimodal function optimization. In J. J. Grenfestette, editor, *Genetic Algorithms and Theri applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49. Lawrence Erlbaum, 1987.

[27] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[28] J. J. Grefenstette. GENESIS : A system for using genetic search precerures. In *Proceedings of the 1984 Conference on intelligent Systems and Machines*, pages 161–165, 1984.

[29] John H. Holland. *Adaptation in Natural an Artificial Systems*. MIT Press, Cambridge, Massachusetts, second edition, 1992.

[30] Jeffrey Horn and Nicholas Nafpliotis. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[31] John Jenkins. *Genetics*. Houghton Mifflin Company, Boston, Massachusetts, 1984.

[32] E.G. Johnson and M.A.G. Abushagur. Micro-Genetic Algorithm Optimization Methods Applied to Dielectric Gratings. *Journal of the Optical Society of America*, 12(5):1152–1160, 1995.

[33] A. K. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptative Systems*. PhD thesis, University of Michigan, Michigan, 1975.

[34] Charles L. Karr. Air-Injected Hydrocyclone Optimization via Genetic Algorithm. In Lawrence Davis, editor, *Handbook of Genetic Algorithms*, pages 222–236. Van Nostrand Reinhold, New York, 1991.

[35] R. N. Keeney and H Raiffa. *Decision with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley & Sons, 1976.

[36] Hajime Kita, Yasuyuki Yabumoto, Naoki Mori, and Yoshikazu Nishikawa. Multi-Objective Optimization by Means of the Thermodynamical Genetic Algorithm. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, Lecture Notes in Computer Science, pages 504–512, Berlin, Germany, September 1996. Springer-Verlag.

[37] Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[38] P Korhonen and J. Laakso. A visual interactive method for solving the multiple-criteria problem. In M. Grauer and A.P. Wierzbicki, editors, *Lecture Notes in Economics and Mathematical Systems 229*, pages 146–153. Springer-Verlag, 1984.

[39] John R. Koza. Herarchical genetic algorithms operating on populations of computer programs. In N. S. Sridharan, editor, *Proceedings of 11th International Joint Conference on Artificial Intelligence*, pages 768–774, San Mateo, California, 1989. Morgan Kaufmann.

[40] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, Massachusetts, 1992.

[41] K. Krishnakumar. Micro-genetic algorithms for stationary and non-stationary function optimization. In *SPIE Proceedings: Intelligent Control and Adaptive Systems*, pages 289–296, 1989.

[42] Frank Kursawe. A variant of evolution strategies for vector optimization. In H. P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature. 1st Workshop, PPSN I*, volume 496 of *Lecture Notes in Computer Science*, pages 193–197, Berlin, Germany, oct 1991. Springer-Verlag.

[43] Zadeh L. Optimality and non-scalar-valued performance criteria. *IEEE transactions on Automatic Control*, 8:59–60, 1963.

[44] Zeleny M. Compromise programming. In M. Zeleny J.L Cochrane, editor, *Mltiple Criteria Decision Making*, pages 262–301, Columbia, South California, 1973. University of South California Press.

[45] Gregor Johann Mendel. Experiments in plant hybridisation. *Journal of Royal Horticultural Society*, 26:1–32, 1901. Traducción al inglés de un artículo publicado.

[46] K Miettinen. *On the Methodology of Multiobjective Optimization with Applications*. PhD thesis, University of Jyväskylä, 1994.

[47] Kaisa M. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer academic publishers, Norwell, Massachusetts 02061 USA, 1999.

[48] M. M. Mäkelä. Issues of implementing a fortran subroutine packae nsolib for nonsmooth optimization. Technical report, University of Jyväskylä, 1993.

[49] H. Nakayama. Sensitivity and trade-off method for problems with multiple kinear fractional objectives an its applications. In A. Lewandowski and V. Volkovich, editors, *Multiobjective Problems of mathematical Programming*, pages 42–50. Lecture Notes in Economics and Mathematical Systems 356, Springer-Verlag, 1991.

[50] A. Osyczka. *Multicreterion Optimization in engineering with FORTRAN programs*. Ellis Horwood Limited, 1984.

[51] Andrezj Osyczka. *Multicriterion Optimization in Engineering with fortran programs*. John Wiley & Sons, 1984.

[52] Yu P.L. A Class of Solutions for Group Desicion Problems. *Management Science*, 19(8):936–946, 1973.

[53] Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart, Alemania, 1973.

[54] R. S. Rosenberg. *Simulation of genetic populations with biochemical properties*. PhD thesis, University of Michigan, Ann Harbor, Michigan, 1967.

[55] Günter Rudolph. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1):96–101, January 1994.

[56] Gass S. and Saaty T. The computational algorithm for the parametric objective function. *Naval Ressearch Logistics Quarterly*, 2:39–45, 1955.

[57] M. Sakawa. Interactive multiobjective decision making by the sequential proxy optimization technique: Spot. *European Journal of Operational Research*, 9(4):386–396, 1982.

[58] J. David Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, 1984.

[59] J. David Schaffer. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.

[60] Hans-Paul Schwefel. Kybernetische evolution als strategie der experimentall forschung in der strömungstechnik. Master's thesis, Universidad Tecnica de Berlin, 1965.

[61] N. Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. Technical report, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India, 1993.

[62] N. Srinivas and Kalyanmoy Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, Fall 1994.

[63] R. E. Steuer. *Multiple Criteria Optimization: Theory, Computation, and Applications*. Number 419–450 in 1. John Wiley & Sons, 1986.

[64] Gilbert Syswerda. Uniform Crossover in Genetic Algorithms. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 2–9, San Mateo, California, 1989. Morgan Kaufmann.

[65] Hisashi Tamaki, Hajime Kita, and Shigenobu Kobayashi. Multi-Objective Optimization by Genetic Algorithms : A Review. In Toshio Fukuda and Takeshi Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation (ICEC'96)*, pages 517–522, Nagoya, Japan, 1996. IEEE.

[66] A. Wetzel. *Evaluation of Efectiveness of genetic algorithms in combinational optimization*. University of Pittsburgh, Pittsburgh (unpublished), 1983.

[67] Darrel Whitley. The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 116–121, San Mateo, California, July 1989. Morgan Kaufmann Publishers.

[68] A. P. Wierzbicki. The ise of reference objectives in multiobjective optimization, multiple criteria decision making theory and applications. In G. Fandel and T. Gal, editors, *Lecture Notes in Economics and Mathematical Systems 177*, pages 468–486, Berlin, Heidelberg, 1980. Springer-Verlag.

[69] August Wismann. *The Germ Plasm: A Theory of Heredity*. Scott, London, UK. United Kingdom, 1893.

[70] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[71] Fengchao Xiao and Hatsuo Yabe. Microwave Imaging of Perfectly Conducting Cylinders from Real Data by Micro Genetic Algorithm Coupled with Deterministic Method. *IEICE Transactions on Electronics*, E81-C(12):1784–1792, December 1998.

[72] L. S. Landon Y. Y. Haimes and D. A. Wismer. On a bicriterion formulation of the problems of itegrated system identification and system optimization. *IEEE Transactions on Systems, Man and Cybernetics*, 1:296–297, 1971.

[73] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.