Algoritmos Evolutivos para Optimización Multiobjetivo: un Estudio Comparativo en un Ambiente Paralelo Asíncrono

Christian von Lücken[1], Augusto Hermosilla[2], Benjamín Barán[3]

Centro Nacional de Computación

Universidad Nacional de Asunción

Campus Universitario de San Lorenzo, Paraguay

P.O. Box 1439, Tel/Fax (595)-21-585550

cvlucken[1], ahermosilla[2], bbaran[3]@cnc.una.py

Resumen

El presente trabajo propone el desarrollo de Algoritmos Evolutivos Multiobjetivos paralelos (*parallel Multi-Objective Evolutionary Algorithms*- pMOEAs). Varios modelos de paralelización alternativos fueron considerados y analizados. Siguiendo distintos modelos propuestos, diferentes MOEAs han sido implementados en paralelo y aplicados en la resolución de problemas de prueba de distinta dificultad. Los resultados obtenidos por los distintos MOEAs, tanto en sus versiones secuenciales como paralelas, han sido comparados y analizados en base a distintas métricas experimentales de desempeño. La paralelización de MOEAs ha demostrado ser una alternativa válida para mejorar el desempeño de los estos algoritmos en todos los problemas de prueba considerados.

Palabras claves: Algoritmos Evolutivos Paralelos, Optimización Multiobjetivo, Frente Pareto.

Dirigido al Workshop de Procesamiento Distribuido y Paralelo (WPDP).

1 Introducción

Los problemas reales usualmente requieren la búsqueda de soluciones que satisfagan en forma simultánea múltiples criterios de desempeño u objetivos los cuales pueden ser contradictorios [1]. Cuando es factible combinar los objetivos de un problema de manera adecuada, es posible considerar un único objetivo a optimizar. En este caso, para obtener *la solución* del problema basta con encontrar el mínimo o el máximo de una única función que resume todos los objetivos que se desean optimizar. Sin embargo, lo usual es que no se conozca la manera óptima de combinar los diferentes objetivos o sea inadecuado, cuando no imposible hacerlo. En este

caso, se dice que el problema es un Problema de Optimización Multiobjetivo (*Multiobjective Optimization Problem* - MOP) [1–4].

En problemas de optimización multiobjetivo con objetivos contradictorios no siempre existe una solución única que pueda ser considerada como la mejor, sino un conjunto de soluciones que representan los mejores compromisos entre los distintos criterios. Dicho conjunto es llamado conjunto Pareto-óptimo y su imagen en el espacio objetivo es denominado Frente Pareto [4].

Los Algoritmos Evolutivos (*Evolutionary Algorithms* - EAs) han demostrado ser especialmente adecuados para la optimización multiobjetivo. La literatura actual reporta un gran número de Algoritmos Evolutivos para Optimización Multiobjetivo (*Multiobjective Evolutionary Algorithms* - MOEA). Con la aplicación cada vez más extendida de MOEAs en problemas reales de optimización, se hace necesario mejorar el desempeño de los mismos a fin de asegurar la aplicabilidad de la técnica en problemas de complejidad creciente. Para ello, una alternativa es la incorporación de conceptos de paralelismo al diseño de estos algoritmos [5, 6].

Si bien las ideas de paralelización de EAs han estado presentes desde prácticamente sus orígenes y existen varios modelos de paralelización para algoritmos evolutivos monobjetivo [7], estos modelos apenas han sido integrados al campo de la optimización evolutiva multiobjetivo [3, 6]. A fin de lograr una mejora en la capacidad de búsqueda de los MOEAs, el presente trabajo propone un marco para el desarrollo y aplicación de algoritmos evolutivos multiobjetivo paralelos (*parallel Multiobjective Evolutionary Algorithms* - pMOEAs) [5,6]. Siguiendo el marco propuesto, distintos pMOEAs han sido desarrollados y aplicados a funciones de prueba de distinta dificultad [8]. A fin de medir el desempeño de los pMOEAs en los distintos problemas multiobjetivo resueltos, se han considerado diferentes métricas experimentales [3, 9].

El presente trabajo está organizado de acuerdo al siguiente esquema: en la sección 2 se introducen los conceptos claves relacionados a la optimización multiobjetivo. En la sección 3 se presentan los MOEAs basados en dominancia Pareto y un modelo de paralelización para el desarrollo de pMOEAs, utilizado en [10]. En base a este modelo de paralelización, las secciones 4 y 5 se presenta una breve descripción de la comparación realizada y los resultados obtenidos. Finalmente se presentan algunas conclusiones en la sección 6.

2 Conceptos básicos y terminología

Mientras que en optimización monobjetivo se busca un vector de decisión n-dimensional que optimice una función escalar, en optimización multiobjetivo se intenta encontrar uno que optimice una función vectorial cuyos elementos representan las distintas funciones objetivo. Un problema de optimización multiobjetivo puede ser definido formalmente como [3]:

Definición 1. *Problema de Optimización Multiobjetivo:* un MOP general optimiza una función

$$\mathbf{y} = \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})) \tag{1}$$

sujeto a

$$\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})) \leq \mathbf{0} \tag{2}$$

donde

$$\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{X} \subseteq \mathbb{R}^n \; ; \; \mathbf{y} = (y_1, \ldots, y_k) \in \mathcal{Y} \subseteq \mathbb{R}^k$$

$\mathbf{x}$ es una variable de decisión vectorial n-dimensional, $\mathbf{y}$ es un vector objetivo k-dimensional, $\mathcal{X} \subseteq \mathbb{R}^n$ denota el espacio de decisión, e $\mathcal{Y} \subseteq \mathbb{R}^k$ denota el espacio objetivo. El conjunto de restricciones dadas por la Ecuación 2 define la región de factibilidad $\mathcal{X}_f \subseteq \mathbb{R}^n$ y cualquier punto $\mathbf{x} \in \mathcal{X}_f$ es una solución factible.

En un MOP con objetivos contradictorios, el espacio de búsqueda se encuentra sólo parcialmente ordenado y dos soluciones pueden ser indiferentes entre sí, siendo poco usual que una única variable de decisión optimice de manera simultánea todos los objetivos. Consecuentemente, para MOPs se extienden los operadores $<$, $>$ e $=$ del siguiente modo:

Definición 2. *Dominancia Pareto:* sean los vectores objetivo $\mathbf{u} = (u_1, \ldots, u_k)$ y $\mathbf{v} = (v_1, \ldots, v_k)$, se dice que $\mathbf{u}$ domina a $\mathbf{v}$ y se denota como $\mathbf{u} \succ \mathbf{v}$ si y sólo si:

$$\forall i \in \{1, \ldots, k\}, u_i \text{ es mejor o igual que } v_i \ \wedge \ \exists i \in \{1, \ldots, k\} | u_i \text{ es mejor que } v_i$$

Es decir, $\mathbf{u}$ domina a $\mathbf{v}$, si $\mathbf{u}$ es mejor o igual a $\mathbf{v}$ en todos los objetivos y estrictamente mejor en al menos un objetivo. Si ninguno de los vectores domina al otro se dice que son indiferentes entre sí ya que ninguno puede ser considerado mejor que el otro considerando todos los objetivos.

Definición 3. Se dice que una solución $\mathbf{x} \in \mathcal{X}_f$ es un Pareto óptimo con respecto a un conjunto $\Omega \subseteq \mathcal{X}_f$ si y sólo si:

$$\nexists \mathbf{x}' \in \Omega \text{ para el cual } \mathbf{v}' = \mathbf{F}(\mathbf{x}') \text{ domina a } \mathbf{u} = \mathbf{F}(\mathbf{x})$$

Cuando resulta claro a que conjunto Ω se hace referencia, simplemente no se lo menciona. La solución $\mathbf{x}$ es Pareto óptima si y sólo si es no-dominada con respecto al conjunto $\mathcal{X}_f$.

Esto es, un vector $\mathbf{x}$ es Pareto óptimo si no existe otro vector de decisión factible $\mathbf{x}' \in \mathcal{X}_f$ que lo domine. El conjunto de soluciones en un MOP, denotado por $\mathcal{P}^\star$, está compuesto por todos estos vectores Pareto óptimos. Las soluciones Pareto óptimas son también llamadas soluciones no inferiores, admisibles, o soluciones eficientes, mientras que sus vectores objetivo correspondientes son denominados *no-dominados*. El conjunto de vectores no-dominados en el espacio objetivo, correspondiente al conjunto de soluciones $\mathcal{P}^\star$, es llamado Frente Pareto Óptimo y denotado por $\mathcal{PF}^\star$ [3].

Definición 4. *Conjunto Pareto Óptimo y Frente Pareto Óptimo:* dado un problema de optimización mutiobjetivo $\mathbf{F}(\mathbf{x})$, el conjunto Pareto óptimo, denotado por $\mathcal{P}^\star$, se define como:

$$\mathcal{P}^\star = \{\mathbf{x} \in \mathcal{X}_f | \ \nexists \mathbf{x}' \in \mathcal{X}_f \text{ para el cual } \mathbf{F}(\mathbf{x}') \text{ domine a } \mathbf{F}(\mathbf{x})\}$$

El Frente Pareto Óptimo $\mathcal{PF}^\star$ correspondiente define como:

$$\mathcal{PF}^\star = \{\mathbf{u} \ = \ \mathbf{F}(\mathbf{x}) | \ \mathbf{x} \in \mathcal{P}^\star\}$$

3 Algoritmos Evolutivos Multiobjetivo Paralelos

Para la mayoría de los MOPs, el conocimiento del Frente Pareto óptimo ayuda al tomador de decisiones a seleccionar aquella solución que representa el mejor compromiso. Generar dicho frente puede ser computacionalmente costoso o incluso imposible, en especial en problemas reales de ingeniería. Entonces, lo único que se puede pretender es obtener una buena aproximación al frente Pareto óptimo verdadero. Los algoritmos evolutivos multiobjetivo son una alternativa práctica en la búsqueda de soluciones de compromiso para problemas reales donde los métodos exactos son inaplicables o ineficientes.

Durante la ejecución de un MOEA basado en dominancia Pareto, un conjunto de soluciones Pareto óptimas con respecto a la población genética actual es encontrado en cada generación. A dicho conjunto se le denomina $P_{known}(t)$, donde t representa el número de generaciones transcurridas desde el inicio del procedimiento evolutivo. El frente Pareto correspondiente a $P_{known}(t)$ se denota como $PF_{known}(t)$. El conjunto de soluciones obtenidas al final de la ejecución de un MOEA basado en Pareto, esto es, el conjunto Pareto conocido, se denota con P_{known}. La notación utilizada para el frente Pareto asociado es PF_{known}. Cuando se resuelve un MOP utilizando MOEAs, la suposición implícita es que se cumple con al menos una de las siguientes relaciones: $PF_{known} = \mathcal{PF}^{\star}$, $PF_{known} \subset \mathcal{PF}^{\star}$ o $PF_{known} \approx \mathcal{PF}^{\star}$ [6], donde $\mathcal{PF}^{\star}$ representa el frente Pareto óptimo teórico real.

La integración de la computación paralela y la computación evolutiva da origen a los algoritmos evolutivos paralelos multiobjetivo (pMOEAs). La utilización de pMOEAs posee varias ventajas con respecto a otros métodos uniendo las características propias de los MOEAs con las ventajas del cómputo paralelo. Básicamente, los pMOEAs intentan encontrar soluciones tan buenas o mejores que sus contrapartes secuenciales en menos tiempo y/o explorar un espacio mayor de posibles soluciones [5, 6, 10].

Básicamente, los modelos de paralelización de EAs más importantes son: el modelo maestro-esclavo, el de difusión y el de islas. En [11] se presenta una revisión de estos modelos aplicados a optimización monobjetivo, mientras que [6] lo hace en un contexto multiobjetivo.

El esquema de paralelización utilizado en el presente trabajo para el desarrollo de MOEAs paralelos se basa en el modelo de islas. Las consideraciones que han conducido a esta elección se detallan en [10]. El modelo de islas está basado en el fenómeno natural de que las poblaciones se encuentran, usualmente, relativamente aisladas unas de otras [11]. En el modelo de islas una población se divide en subpoblaciones separadas e independientes, llamadas islas. Los diferentes operadores evolutivos trabajan en cada isla, lo que implica que las distintas poblaciones se encuentran explorando en regiones diferentes del espacio de búsqueda. Cada isla también podría tener distintos parámetros así como diferente estructura de MOEA. Además, individuos de una isla podrían migrar a otra isla de acuerdo a algún criterio.

El modelo de islas requiere la selección de una política de migración que señale: la manera en que los individuos migrarán, el número de migrantes, la frecuencia de migración, de dónde se seleccionarán los elementos a migrar y cómo se realizará el reemplazo de los elementos en una población por los migrantes provenientes de otras poblaciones. Además, es preciso definir los distintos parámetros y algoritmos a utilizar en cada una de las diferentes islas.

Definiendo convenientemente la política de migración, el modelo de islas puede aplicarse

a varias arquitecturas paralelas, especialmente en sistemas paralelos de memoria distribuida. Primeramente, se establece un criterio para la selección de soluciones a enviar. En este trabajo se ha decidido seleccionar como migrantes los mejores individuos de una población en el momento que la migración ocurre. Un parámetro limita el número máximo de soluciones a enviar [10].

Además, se define un criterio que determinará la frecuencia en que ocurrirá la migración. Se han explorado varias opciones como: enviar individuos transcurrido un número de generaciones, enviar individuos de forma probabilística, enviar individuos de forma adaptativa, entre otras [10]. En el primer caso, se determina el número de generaciones que deben transcurrir para que se produzca un envío. En el segundo, se proporciona como parámetro del algoritmo la probabilidad que un envío ocurra en una determinada generación. La tercera opción, se basa en modificar la frecuencia de migración conforme el proceso de evolución ocurre [5].

En el presente trabajo, tanto el envío como la recepción de las soluciones se maneja en forma asíncrona. Esto permite que los MOEAs puedan continuar inmediatamente después que han ejecutado una primitiva de envío así como la verificación, sin espera, de la recepción de soluciones. Además de la reducción en el tiempo de espera, la comunicación asíncrona es adecuada para lidiar con la posible pérdida de datos y elementos de procesamiento. En cuanto a la forma en que procederá el reemplazo de elementos de la población por migrantes, habiendo explorado varias opciones, se ha optado por el reemplazo aleatorio de elementos dominados por elementos recibidos [10].

En este trabajo se presentan pMOEAs con topología de migración descentralizada, el cual se basa en la utilización de dos tipos distintos de procesos formando un equipo de trabajo compuesto por un proceso colector y p procesos MOEAs (Algoritmo 1).

Durante la evolución, cada uno de los MOEAs que interviene en la búsqueda envía a todos los elementos que componen el equipo de trabajo un porcentaje de las mejores soluciones obtenidas. El colector almacena las soluciones recibidas y se eliminan las soluciones cubiertas (dominadas o iguales). En caso que el número de elementos en el conjunto de soluciones del colector supere un máximo de soluciones deseadas, se procede a reducir el tamaño del conjunto utilizando el procedimiento de *clustering* del *Strenght Pareto Evolutionary Algorithm* (SPEA) [1]. Cuando finalizan todos los MOEAs, se eliminan los elementos cubiertos en el conjunto de soluciones del colector y se devuelven los resultados obtenidos.

El Algoritmo 1 presenta el marco general propuesto para la implementación de los distintos pMOEAs.

4 Comparación Experimental de pMOEAs

4.1 Algoritmos implementados

Los MOEAs se pueden clasificar de acuerdo a si incluyen o no mecanismos para la preservación de las buenas soluciones encontradas durante la evolución (elitismo) [4], en MOEAs de primera y segunda generación. A fin de determinar la mejora en el desempeño que se puede obtener con la utilización de pMOEAs y considerando la falta de estudios experimentales que analicen las distintas cuestiones relativas al desarrollo de los mismos y su desempeño con relación a los

Algoritmo 1 Procedimiento MOEA paralelo general.

Procedimiento: pMOEA
Recibir parámetros usuales del MOEA, así como la probabilidad de migración p_{mig} y el número de elementos no-dominados a migrar n_{mig}.
Unirse al equipo de trabajo
Generar la población inicial $P(0)$ de forma aleatoria y hacer $t = 0$
repeat
  if se han recibido soluciones provenientes de otros procesos **then**
    Almacenar las soluciones recibidas en P_{recv}
    Reemplazar elementos de $P(t)$ por elementos en P_{recv}
  end if
  Generar nueva población $P(t + 1)$
  $t = t + 1$
  if Se cumple criterio de envío **then**
    Seleccionar elementos a migrar de $P(t)$ en P_{mig}
    Enviar P_{mig} a todos los elementos en el equipo de trabajo
  end if
until Alcanzar el criterio de parada
Enviar $P_{known}(t)$ al colector con señal de finalización
Salir del grupo de trabajo y terminar ejecución

algoritmos secuenciales, en este trabajo se han paralelizado siguiendo el modelo propuesto, seis algoritmos evolutivos representativos de las dos generaciones de MOEAs señaladas.

Los MOEAs representativos de la primera generación implementados son: el *Multiobjective Genetic Algorithm* (MOGA) [12], el *Niched Pareto Genetic Algorithm* (NPGA) [13] y el *Non-dominated Sorting Genetic Algorithm* (NSGA) [14]. Por otro lado, MOEAs representativos de la segunda generación paralelizados en este trabajo son: el SPEA [9], el NSGA-II [15], y el *Controlled Elitist NSGA-II* (CNSGA-II) [16]. Estos algoritmos fueron seleccionados considerando su utilización en trabajos previos de comparación [5, 15, 17].

Para comparar el desempeño de los distintos MOEAs paralelizados, primeramente se ha utilizado un conjunto de funciones de prueba [1]. Los resultados obtenidos se han comparado utilizando distintas métricas experimentales [3, 17].

4.2 Funciones de prueba ZDT

En base a diferentes características que pueden hacer que los MOEAs tengan dificultades para converger al conjunto Pareto óptimo y mantener la diversidad en la población, identificadas en [18], en [1] se presenta un conjunto de funciones de prueba que contempla las distintas posibilidades existentes. Las mismas son conocidas como las funciones ZDT.

A fin de comparar los distintos algoritmos con respecto a la convexidad, la no-convexidad, la discretitud y la multimodalidad, en el presente trabajo se utilizan cuatro de estas funciones: ZDT1, ..., ZDT4 [1]. Dichas funciones consideran la minimización de dos objetivos y cada una

Función	m	x_i	$g(x_2, \ldots, x_m)$	$h(f_1(x_1), g(x_2, \ldots, x_m))$
ZDT1	30	$x_i \in [0,1]$	$1 + 9\sum_{i=2}^{m} \frac{x_i}{(m-1)}$	$1 - \sqrt{\frac{f_1}{g}}$
ZDT2	30	$x_i \in [0,1]$	$1 + 9\sum_{i=2}^{m} \frac{x_i}{(m-1)}$	$1 - (\frac{f_1}{g})^2$
ZDT3	30	$x_i \in [0,1]$	$1 + 9\sum_{i=2}^{m} \frac{x_i}{(m-1)}$	$1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi f_1)$
ZDT4	10	$x_1 \in [0,1]$ $x_2, \ldots, x_m \in [-5,5]$	$1 + 10(m-1) + \sum_{i=2}^{m}(x_i^2 - 10\cos(4\pi x_i))$	$1 - \sqrt{\frac{f_1}{g}}$

Tabla 1: Características de las funciones de prueba utilizadas.

de ellas está estructurada de la misma forma:

$$\begin{aligned} &\text{Minimizar } F(\mathbf{x}) = (f_1(x_1), f_2(\mathbf{x})) \\ &\text{sujeto a: } f_2(\mathbf{x}) = g(x_2, \ldots, x_m)h(f_1(x_1), g(x_2, \ldots, x_m)) \\ &\text{donde: } \mathbf{x} = (x_1, \ldots, x_m) \end{aligned} \tag{3}$$

La función f_1 es una función que depende únicamente de la primera variable de decisión, definida para todas las funciones de prueba consideradas como: $f_1(x_1) = x_1$, donde $x_1 \in [0,1]$. Así, las funciones de prueba seleccionadas difieren en la definición de g y h, así como en el número de variables m y en los valores que éstas pueden tomar. La Tabla 1 presenta estos valores.

4.3 Métricas de desempeño

En optimización multiobjetivo no existe un criterio único que permita establecer con facilidad si un conjunto de aproximación a $\mathcal{PF}^\star$ es mejor que otro puesto que se pueden considerar varias medidas de calidad, por ejemplo: la cercanía al conjunto Pareto óptimo, el número de soluciones obtenidas, la distribución de las soluciones, representan métricas alternativas [3].

Para comparar los resultados obtenidos por distintos algoritmos evolutivos se han desarrollado varias métricas de desempeño, las cuales buscan capturar las características que hacen a una aproximación del conjunto Pareto-óptimo mejor que otra en algún criterio. Por lo general, éstas asignan a un conjunto aproximación particular un número que refleja un aspecto de calidad determinado [3]. Otras métricas utilizadas para determinar si un conjunto aproximación es mejor que otro son binarias y asignan a un par de conjuntos a comparar un número que indica como éstos se relacionan [9].

En este trabajo se han utilizado ocho métricas distintas evaluando distintas características del conjunto aproximación. Explicaciones sobre cada una de las métricas seleccionadas puede ser encontrada en [3, 9]. La Tabla 2, resume los aspectos más importantes de las mismas.

4.4 Parámetros utilizados

Fuera de las modificaciones propias para la implementación en paralelo, los MOEAs utilizados fueron implementados según la literatura original de referencia para la resolución de los distintos

Métrica	**Descripción**	**Valor ideal**
Generación Total de Vectores No-dominados (ONVG)	Mide el número de soluciones en PF_{known}	Máximo posible
Razón de la Generación Total de Vectores No-dominados (ONVGR)	Indica la razón entre la cantidad de vectores no-dominados encontrados por cada algoritmo y la cantidad de vectores no-dominados existentes en $\mathcal{PF}^{\star}$	ONVGR = 1
Número de soluciones no-dominadas verdaderas (N)	Es la cantidad de vectores no-dominados propuestos que en efecto pertenecen al frente Pareto óptimo real.	Máximo posible
Razón del error (E)	Reporta la proporción de vectores objetivo en PF_{known} que no son miembros de $\mathcal{PF}^{\star}$	E = 0
Distancia Generacional (GD)	Representa que tan lejos se encuentra PF_{known} de $\mathcal{PF}^{\star}$	GD = 0
Error Máximo del Frente Pareto (ME)	Indica una banda de error máxima cuando se considera PF_{known} con respecto a $\mathcal{PF}^{\star}$.	ME= 0
Espaciamiento (S)	Sirve como un indicador de la distribución de las soluciones en PF_{known}	S= 0
Cobertura (C)	Representa la relación del número de vectores en el espacio objetivo, encontrados por un algoritmo dado, que son mejores que los encontrados por otro	En promedio, el menor valor de soluciones cubiertas y el mayor valor de cobertura [10]

Tabla 2: Métricas utilizadas.

problemas considerados. Para la obtención de los resultados presentados en las siguientes secciones, las implementaciones paralelas se realizaron utilizando el marco general propuesto en la sección 3. En todos los casos los operadores de mutación y cruzamiento utilizados fueron mutación de un sólo bit y cruzamiento de un sólo punto. Las soluciones de las distintas funciones de prueba se codificaron como cadenas binarias.

Todos los programas se escribieron en lenguaje C y se compilaron utilizando gcc. Para las implementaciones paralelas se utilizaron las primitivas de comunicación proveídas por PVM [19]. El entorno computacional para las distintas corridas estuvo constituido por 20 computadoras AMD K6-2 de 700MHz con 128 MB de memoria RAM y utilizando Red Hat Linux v7.3.

Para la resolución de los distintos problemas de prueba se han utilizado pMOEAs homogéneos [6], es decir utilizando el mismo MOEA en cada isla y con parametros iguales difiriendo sólo en la semilla utilizada para la generación de la población inicial. Estos parámetros, según corresponda a cada MOEA utilizado, se presentan en la Tabla 3.

Para los problemas ZDT considerados se realizaron 10 ejecuciones de cada implementación secuencial de los 6 algoritmos utilizados, y 10 ejecuciones de las versiones paralelas utilizando 1, 2 y 4 procesos pMOEAs. Para las diferentes corridas, se han utilizado diferentes semillas

para la generación de números aleatorios. En las distintas corridas paralelas presentadas la migración procede con una probabilidad (p_{mig}) igual a 0.5 por generación. En cada migración, el número máximo de soluciones intercambiadas entre los distintos procesos (n_{mig}) es 10.

Para el cálculo de las métricas que requieren el frente Pareto óptimo $\mathcal{PF}^{\star}$ se utilizó, para cada problema, una aproximación PF_{true} obtenida mediante la unión de todos los resultados obtenidos considerando todas las ejecuciones realizadas, de donde se eliminaron las soluciones dominadas.

5 Resultados experimentales y análisis

Los resultados de las distintas ejecuciones se agruparon de acuerdo al número de elementos de proceso y algoritmo utilizado obteniéndose un conjunto de soluciones nodominadas para cada conjunto formado. Por simplicidad, a la corrida secuencial se nombró como corrida a, mientras que, a las corridas paralelas que utilizan 1, 2 y 4 procesos pMOEA como corrida b, c y d respectivamente. Al conjunto de soluciones propuesto por la corrida a (secuencial) que utiliza al SPEA como algoritmo evolutivo se nombró como SPEA-a, mientras que el conjunto obtenido utilizando el NSGA como NSGA-a, y de forma similar para los demás algoritmos y corridas consideradas.

Siendo la comparación entre MOEAs en sí misma multiobjetivo, a fin de resumir los resultados obtenidos en [10], se considera un ordenamiento lexicográfico de las mismas de acuerdo a: la clasificación por no-dominancia (*ranking*) [2] considerando todas las métricas de forma simultánea; y el valor de la métrica N. Estos objetivos se han considerado como una alternativa que representa el caso en el cual el tomador de decisiones está interesado en N como *la* métrica de desempate, pudiendo existir otras opciones como por ejemplo S cuando se desea un frente uniformemente distribuido.

Es interesante notar que son las implementaciones basadas en SPEA y en el NSGA-II con cuatro procesadores las que se ubican en los primeros lugares dependiendo de la función de prueba considerada. Otra cuestión importante es que no necesariamente el algoritmo que obtiene en mejor desempeño cuando se comparan las implementacíones secuenciales es el que obtiene la mejor clasificación en sus versiones paralelas. Además, entre las implementaciones de primera generación son las del NSGA las que se encuentran mejor posicionadas.

	Parámetro						
Algoritmo	p_m	p_c	N	N'	σ_{share}	t_{dom}	t_{red}
CNSGA-II	0.01	0.8	100	-	-	-	0.7
NSGA-II	0.01	0.8	100	-	-	-	-
SPEA	0.01	0.8	100	100	-	-	-
NSGA	0.01	0.8	100	-	0.41	-	-
NPGA	0.01	0.8	100	-	0.41	10%	-
FFGA	0.01	0.8	100	-	0.41	-	-

Tabla 3: Parámetros utilizados en las diferentes corridas.

ZDT1			ZDT2			ZDT3			ZDT4		
Conjunto	**Pos.**	**N**	**Conjunto**	**Pos.**	**N**	**Conjunto**	**Pos.**	**N**	**Conjunto**	**Pos.**	**N**
NSGA-II-d	1	1	SPEA-d	1	1	NSGA-II-d	1	1	SPEA-d	1	1
SPEA-d	2	2	NSGA-II-d	1	2	NSGA-II-c	1	2	CNSGA-II-d	1	2
NSGA-II-c	2	3	CNSGA-II-d	1	3	SPEA-d	1	3	SPEA-c	1	3
CNSGA-II-d	3	4	SPEA-c	2	4	CNSGA-II-d	1	4	CNSGA-II-c	1	4
SPEA-c	4	5	NSGA-II-c	2	6	SPEA-c	1	5	NSGA-II-d	1	5
CNSGA-II-c	4	6	CNSGA-II-c	3	5	NSGA-II-b	1	6	SPEA-b	1	6
SPEA-b	5	7	NSGA-II-b	3	8	CNSGA-II-c	1	7	NSGA-d	1	7
SPEA-a	5	8	NSGA-II-a	3	9	SPEA-a	1	9	SPEA-a	1	8
NSGA-II-b	5	9	SPEA-b	4	7	CNSGA-II-b	1	10	NSGA-II-c	1	9
NSGA-II-a	5	10	CNSGA-II-b	4	10	NSGA-d	1	13	NSGA-II-a	1	10
CNSGA-II-b	6	11	SPEA-a	4	11	FFGA-a	1	21	NSGA-II-b	1	11
CNSGA-II-a	6	12	FFGA-c	4	13	NSGA-II-a	2	8	CNSGA-II-a	1	12
NSGA-d	7	13	CNSGA-II-a	5	12	SPEA-b	2	11	NPGA-d	1	13
FFGA-d	7	14	FFGA-d	6	13	CNSGA-II-a	2	12	NPGA-b	1	13
NSGA-a	7	16	NSGA-c	6	14	NSGA-c	3	14	NPGA-c	1	14
NSGA-c	8	14	NSGA-d	6	15	NSGA-b	3	15	NSGA-c	1	14
FFGA-c	8	15	FFGA-b	6	16	NSGA-a	3	16	CNSGA-II-b	1	15
NSGA-b	9	15	NSGA-b	7	15	NPGA-b	3	19	FFGA-a	1	17
NPGA-d	10	16	NSGA-a	7	16	FFGA-d	3	20	NPGA-a	1	18
NPGA-c	11	16	NPGA-d	7	16	FFGA-c	3	24	NSGA-a	1	18
FFGA-b	11	16	NPGA-c	7	16	NPGA-d	4	17	FFGA-c	1	19
NPGA-b	11	16	NPGA-b	8	16	NPGA-c	4	18	FFGA-b	1	20
NPGA-a	11	16	NPGA-a	8	17	NPGA-a	4	22	NSGA-b	1	20
FFGA-a	12	16	FFGA-a	9	17	FFGA-b	4	23	FFGA-d	2	16

Tabla 4: Orden lexicográfico de los MOEAs para cada problema.

6 Conclusiones

El diseño e implementación de algoritmos evolutivos multiobjetivo paralelos es un problema complejo. Se hace necesario un mayor estudio sobre las diferentes alternativas de paralelización existentes y la determinación de parámetros adecuados de migración.

En [10] se puede ver que mientras los algoritmos elitistas obtienen un conjunto de soluciones cercanas al frente Pareto óptimo con una mejor distribución de soluciones, los no elitistas apenas obtienen soluciones en dicho frente.

De acuerdo con los resultados obtenidos, se puede establecer que los algoritmos de segunda generación implementados han sido los que lograron, en general, un mejor desempeño. Además, en la mayoría de los problemas analizados se ha encontrado que los algoritmos CNSGA-II y NSGA-II son los más beneficiados con la paralelización, utilizando el modelo propuesto.

Al utilizar pMOEAs, se extiende el espacio de búsqueda. La recepción de elementos provenientes de distintas islas introduce información genética en forma aleatoria que es útil para la obtención de mejores soluciones. Es necesario determinar la mejor manera de incorporar esta información en las diferentes islas.

En resumen a partir de los distintos resultados presentados queda claro que la utilización de pMOEAs es adecuada para la búsqueda de soluciones en espacios de búsqueda complejos y de alta dimensionalidad. Además, se ha establecido la importancia de la utilización de MOEAs basados en elitismo para la implementación de pMOEAs.

Entre los MOEAs utilizados se recomienda para el desarrollo de aplicaciones paralelas la

utilización del NSGA-II ya que es, en la mayoría de los problemas, el más beneficiado con la paralelización, esto a pesar que las implementaciones secuenciales del SPEA, en general, son mejores que las del NSGA-II.

Como trabajos futuros se proponen, entre otros, los siguientes: incorporar otros enfoques al conjunto de MOEAs comparados; comparar el desempeño de los distintos MOEAs utilizando una política diferente de migración y realizar comparaciones utilizando implementaciones paralelas que utilicen distintos algoritmos evolutivos simultáneamente, formando un equipo o *team algorithm* [20].

Referencias

[1] E. Zitzler, K. Deb, y L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, vol. 8, pg. 173–195, Verano 2000.

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

[3] D. A. van Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. Tesis Doctoral, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Ohio, Mayo 1999.

[4] C. A. Coello Coello, "A Short Tutorial on Evolutionary Multiobjective Optimization," *First International Conference on Evolutionary Multi-Criterion Optimization* (E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, y D. Corne, eds.), pg. 21–40, Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[5] C. von Lücken, B. Barán, y A. Sotelo, "Pump schedulling optimisation using parallel multiobjective evolutionary algorithms," *XXVII Conferencia Latinoamericana de Informática CLEI-2003*, (La Paz, Bolivia), 2003.

[6] D. A. van Veldhuizen, J. B. Zydallis, y G. B. Lamont, "Considerations in Engineering Parallel Multiobjective Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pg. 144–173, 2003.

[7] E. Cantú-Paz, "A summary of research on parallel genetic algorithms," Tech. Rep. 95007, Dep. of General Engineering, Univ. of Illinois at Urbana-Champaign, Urbana, IL, 1999.

[8] E. Zitzler, K. Deb, y L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms on Test Functions of Different Difficulty," *Proceedings of the 1999 Genetic and Evolutionary Computation Conference.*, (Orlando, Florida), pg. 121–122, Julio 1999.

[9] E. Zitzler y L. Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, pg. 257–271, Noviembre 1999.

[10] C. von Lücken, "Algoritmos evolutivos para optimización multi-objetivo:un estudio comparativo en un ambiente paralelo asíncrono," Tesis de Maestría, Universidad Nacional de Asunción, San Lorenzo, Paraguay, Diciembre 2003.

[11] E. Cantu-Paz, "A survey of parallel genetic algorithms," *Calculateurs Paralleles, Reseaux et Systemes Repartis*, vol. 10, no. 2, pg. 141–171, 1998.

[12] C. M. Fonseca y P. J. Fleming, "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," *Proc. of the Fifth International Conference on Genetic Algorithms* (S. Forrest, ed.), (San Mateo, California), University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers, 1993.

[13] J. Horn y N. Nafpliotis, "Multiobjective Optimization using the Niched Pareto Genetic Algorithm," Tech. Rep. IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[14] N. Srinivas y K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," tech. rep., Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India, 1993.

[15] K. Deb, S. Agrawal, A. Pratab, y 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.

[16] K. Deb y T. Goel, "Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence," *First International Conference on Evolutionary Multi-Criterion Optimization* (E. Z. et al., ed.), pg. 67–81, Springer-Verlag. LNCS No. 1993, 2001.

[17] E. Zitzler, *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications.* Tesis Doctoral, Instituto Federal de Tecnología de Suiza (ETH), Zurich, Switzerland, Noviembre 1999.

[18] K. Deb, "Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems," Tech. Rep. CI-49/98, Dortmund: Department of Computer Science/LS11, University of Dortmund, Germany, 1998.

[19] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, y V. Sunderam, *PVM:Paralell Virtual machine - A user's guide and Tutorial for Networked parallel Computing.* Cambridge, MA: M.I.T. press, 1994.

[20] B. Barán, E. Kaszkurewicz, y A. Bhaya, "Parallel asynchronous team algorithms: Convergence and performance analysis," *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, no. 7, pg. 677–688, 1996.