Una Versión Paralela del Algoritmo Evolutivo para Optimización Multiobjetivo NSGA-II

Sergio Nesmachnow

sergion@fing.edu.uy
Centro de Cálculo, Instituto de Computación
Facultad de Ingeniería.
Universidad de la República, Uruguay.

Resumen

Este trabajo presenta una versión paralela del algoritmo evolutivo para optimización multiobjetivo NSGA-II (*Non-dominated Sorting Genetic Algorithm*, versión II), original de Deb, Agrawal, Pratab y Meyarivan (2000). Se introducen los detalles de diseño e implementación de una versión paralela basada en subpoblaciones semi-independientes y se analiza la calidad de resultados y la eficiencia computacional, comparando con los resultados y tiempos de ejecución de la versión secuencial del algoritmo NSGA-II sobre un conjunto de problemas de prueba estándar.

Palabras clave: Algoritmos Evolutivos Paralelos, Optimización Multiobjetivo, NSGA-II.

Destinado al Workshop de Agentes y Sistemas Inteligentes.

I. INTRODUCCIÓN

Los *Algoritmos Evolutivos* (EAs) se han popularizado como métodos robustos y efectivos para la resolución de problemas de optimización. Tradicionalmente, los problemas abordados consideran la optimización de una única función objetivo, pero en la última década se han desarrollado varios EAs para afrontar problemas con objetivos múltiples, que tienen características que los diferencian de los EAs tradicionales.

Las técnicas de procesamiento paralelo se aplican a los EAs con el objetivo de mejorar la eficiencia computacional y perfeccionar el mecanismo evolutivo (Cantú-Paz, 2001). Desde la perspectiva de la eficiencia, paralelizar un EA permite acelerar la resolución de problemas cuya dimensión motiva utilizar poblaciones numerosas o evaluar complejas funciones objetivo. Desde el punto de vista algorítmico, los modelos paralelos pueden explotar el paralelismo intrínseco del mecanismo evolutivo, trabajando simultáneamente sobre varias poblaciones independientes para resolver un problema.

En el área de la optimización multiobjetivo, pocos trabajos han abordado el estudio del paralelismo aplicado y sus influencias en la eficiencia computacional y calidad de soluciones de los algoritmos. Este trabajo propone el estudio de un modelo de paralelismo aplicado al conocido algoritmo evolutivo para optimización multiobjetivo NSGA-II original de Deb et al. (2000). El esquema de paralelismo aplicado corresponde a un modelo de poblaciones múltiples y migración.

El resto del documento se organiza del modo que se describe a continuación: la sección 2 introduce conceptos vinculados con los problemas de optimización multiobjetivo (MOPs). La sección 3 presenta conceptos básicos sobre los algoritmos evolutivos y su aplicación a los MOPs, comentando el algoritmo NSGA-II. Asimismo, se explica la aplicación de las técnicas de procesamiento paralelo a los EAs en general y se reseñan propuestas existentes de paralelismo aplicado al NSGA-II. Los detalles de diseño e implementación de la versión paralela diseñada se presentan en la sección 4. El conjunto de problemas de prueba y las métricas utilizadas para evaluar la eficiencia y la calidad de las soluciones obtenidas se presentan en la sección 5. La sección 6 presenta y analiza los resultados experimentales. Por último, la sección 7 ofrece las conclusiones del trabajo y líneas de trabajo futuro.

II. PROBLEMAS DE OPTIMIZACIÓN MULTIOBJETIVO

Esta sección presenta una breve introducción a los MOPs y conceptos relacionados.

Un MOP plantea optimizar (minimizar o maximizar) un conjunto de funciones, habitualmente en conflicto entre sí. La existencia de múltiples funciones objetivo plantea una diferencia fundamental con un problema monoobjetivo: no existe una *única* solución al problema, sino un conjunto de soluciones que plantean diferentes compromisos entre los valores de las funciones a optimizar. La Figura 1 presenta la formulación general de un MOP. Gran parte de los problemas de optimización subyacentes a problemas del mundo real tienen una formulación de este tipo, aunque en general son abordados en la práctica mediante un enfoque monoobjetivo.

Minimizar / Maximizar	$\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \mathbf{K}, f_M(\mathbf{x}))$
sujeto a	$\mathbf{G}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \mathbf{K}, g_S(\mathbf{x})) \geq \mathbf{0}$
	$\mathbf{H}(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}), \mathbf{K}, h_R(\mathbf{x})) = \mathbf{0}$
	$x_i^{(L)} \leq x_i \leq x_i^{(U)} \quad 1 \leq i \leq N$

Figura 1: Problema de Optimización Multiobjetivo.

Siendo Ω la región factible del problema, determinada por las restricciones impuestas por las funciones **G** y **H**, la solución al MOP corresponde a un vector de variables de decisión $\mathbf{x}=(x_1,x_2,\mathbf{K},x_N)\in\Omega$, cuyos valores representen un compromiso adecuado para las funciones $f_1,f_2,\mathbf{K},f_M$. Considerando el caso de minimización de funciones, un punto $\mathbf{x}^*$ es llamado *óptimo de Pareto* si $\forall\,\mathbf{x}\in\Omega$ se cumple que $f_i(\mathbf{x})=f_i(\mathbf{x}^*)\ \forall i\in\{1,...,M\}$ o para al menos un valor de i $f_i(\mathbf{x})>f_i(\mathbf{x}^*)$. Esto significa que no existe un vector factible que sea *mejor* que el óptimo de Pareto en una función objetivo sin ser peor en los valores de alguna de las restantes funciones. Asociada a esta definición se introduce la relación de *dominancia*, un orden parcial entre soluciones del MOP; un vector $\mathbf{w}=(w_1,w_2,\mathbf{K},w_N)$ *domina* a otro $\mathbf{v}=(v_1,v_2,\mathbf{K},v_N)$ (se nota $w\ \mathbf{p}\ v$) si $w_i\le v_i\ \forall i\in\{1,...,M\}\wedge\exists i\in\{1,...,M\}/w_i<v_i$. Dado que diferentes valores de las variables representan diferentes compromisos entre los valores de las funciones, la resolución de un MOP no se concentra en hallar un único valor solución, sino en hallar un conjunto de soluciones *no dominadas*, de acuerdo a la definición presentada anteriormente.

El conjunto de soluciones óptimas del MOP está compuesto por los vectores factibles no dominados, se le denomina *conjunto óptimo de Pareto* y queda definido por $P^*=\{\mathbf{x}\in\Omega/\neg\exists\,\mathbf{x}'\in\Omega\ f(\mathbf{x}')\ \mathbf{p}\ f(\mathbf{x})\}$. La región de puntos definida por el conjunto óptimo de Pareto en el espacio de valores de las funciones objetivo se conoce como *frente de Pareto*. Formalmente, el frente de Pareto está definido por $FP^*=\{\mathbf{u}=(f_1(\mathbf{x}),f_2(\mathbf{x}),\mathbf{K},f_M(\mathbf{x}))/\mathbf{x}\in P^*\}$.

III. Algoritmos Evolutivos Para Optimización Multiobjetivo

La complejidad inherente de los MOPs plantea un difícil reto para los algoritmos clásicos como las técnicas enumerativas o métodos de búsqueda local, basados en gradientes o que utilizan técnicas determinísticas –*greedy, branch & bound*, etc. Estos métodos tradicionales exigen elevados costos computacionales para resolver MOPs complejos, sobre espacios de soluciones de dimensiones elevadas. En este contexto, las *técnicas heurísticas estocásticas* se han propuesto como alternativas para alcanzar soluciones aproximadas de buena calidad en tiempos de cómputo razonables. Dentro de esta categoría, las técnicas de computación evolutiva se han manifestado como métodos robustos y efectivos para resolver los MOPs y se han popularizado a consecuencia de su éxito.

Los *Algoritmos Evolutivos para Optimización Multiobjetivo* (MOEAs) surgen como una extensión de los EAs para problemas monoobjetivo, utilizando conceptos relacionados con el tratamiento de funciones multimodales. Esta sección introduce los EAs y sus variantes aplicadas a la resolución de MOPs, examinando el algoritmo NSGA-II. Se presentan las ideas sobre la aplicación de técnicas de paralelismo a los EAs, y se resumen brevemente las propuestas existentes de paralelismo aplicadas al algoritmo NSGA-II.

A. Algoritmos Evolutivos

Los EAs basan su funcionamiento en la simulación del proceso de evolución natural de los seres vivos (Goldberg, 1989). Consisten en una técnica que aplica operadores estocásticos sobre un conjunto de individuos –la población– con el propósito de mejorar su *fitness*, una medida relacionada con la función objetivo del problema. Cada individuo representa una solución potencial del problema, codificada de acuerdo a un esquema de representación.

La población se genera aleatoriamente, y evoluciona aplicando iterativamente *operadores de reproducción*: recombinaciones de individuos –*cruzamientos*– y modificaciones aleatorias –*mutaciones*–. La evolución es guiada por una estrategia de selección de los individuos más adaptados a la resolución del problema, de acuerdo a sus valores de *fitness*.

B. Algoritmos Evolutivos para Optimización Multiobjetivo

La capacidad de los EAs para resolver problemas con objetivos múltiples fue sugerida en 1967 por Rosenberg, pero recién en 1984 Schaffer presentó el primer MOEA. A partir de 1990 se realizaron numerosas propuestas de MOEAs, formándose una comunidad de investigadores en el área que trabaja activamente en la actualidad (Coello et al, 2002)..

Al trabajar sobre un conjunto de soluciones, los EAs son capaces de resolver problemas con objetivos múltiples. En cada ejecución los EAs hallan un conjunto de soluciones aproximadas al frente de Pareto, mientras que los algoritmos tradicionales sólo generan una solución por ejecución. Además, los EAs son menos sensibles a la forma y continuidad del frente de Pareto y permiten abordar problemas de grandes dimensiones.

Un MOEA debe diseñarse para lograr dos propósitos simultáneamente: aproximarse al frente de Pareto y mantener la diversidad de las soluciones, muestreando adecuadamente el espacio de soluciones para no converger a una solución única o a una sección del frente. La Figura 2 presenta los propósitos de un MOEA, donde se ha marcado con la región celeste (gris) el espacio de funciones objetivo de un problema hipotético, mientras que la línea gruesa roja (oscura) representa al frente de Pareto. El mecanismo evolutivo permite lograr el primer propósito, mientras que para preservar la diversidad los MOEAs aplican técnicas empleadas tradicionalmente por los EAs en la optimización de funciones multimodales (*nichos*, *sharing*, *crowding*, etc.).

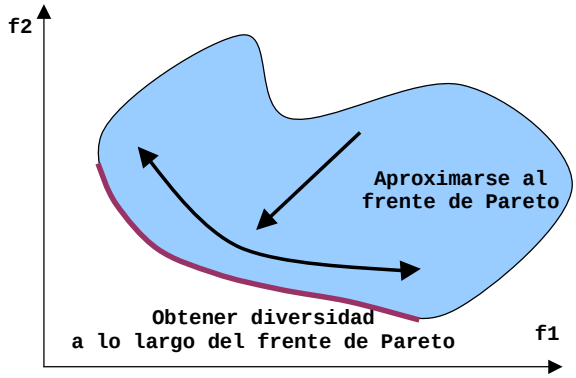

Figura 2: Propósitos de un Algoritmo Evolutivo para Optimización Multiobjetivo.

La Figura 3 presenta un esquema genérico para un MOEA. Es posible observar dos operadores característicos de un MOEA, que no aparecen en un EA monoobjetivo: el `Operador de diversidad` que aplica una técnica para evitar la convergencia puntual a un sector del frente de Pareto y un procedimiento para `Asignar fitness` que brinda mayor chance de perpetuarse a individuos con mejores características, considerando los valores de las funciones objetivo del problema y eventualmente una métrica utilizada para evaluar la diversidad de soluciones.

```
Inicializar(Poblacion(0)); generacion = 0
mientras (no CriterioParada) hacer
  Evaluar(Poblacion(generacion))
  Operador de diversidad(Poblacion(generacion))
  Asignar fitness(Poblacion(generacion))
  Padres = Seleccion(Poblacion(generacion))
  Hijos = Operadores de Reproduccion(Padres)
  NuevaPop = Reemplazar(Hijos,Poblacion(generacion))
  generacion ++
  Poblacion(generacion) = NuevaPop
retornar Mejor Solucion Hallada
```

Figura 3: Esquema de un Algoritmo Evolutivo para Optimización Multiobjetivo.

C. El Algoritmo NSGA-II

El algoritmo NSGA-II (Deb et al, 2000) fue propuesto para resolver tres aspectos fuertemente criticados en el algoritmo NSGA original (Srinivas y Deb, 1994): la ausencia de elitismo, el ordenamiento no dominado, y la dependencia del parámetro σ de *sharing*.

La Figura 4 ofrece un esquema del NSGA-II, presentando sus características principales: el ordenamiento no dominado elitista usando una población auxiliar, que reduce la complejidad de los chequeos de dominancia de $\mathbf{O}(MP^3)$ *a* $\mathbf{O}(MP^2)$, siendo *M* el número de funciones objetivo y *P* el tamaño de la población; la preservación de diversidad calculando distancias de *crowding* y la asignación de fitness que se basa en rangos de no dominancia, incorporando los valores de distancia de *crowding* usados para evaluar la diversidad.

```
Inicializar(Poblacion(0)); generacion = 0
mientras (no CriterioParada) hacer
    Evaluar(Poblacion(generacion))
    R = Padres ∪ Hijos
    Frentes = Ordenamiento No Dominado(R)
    NuevaPop = ∅; i=1
    mientras |NuevaPop| + |Frentes(i)| ≤ sizepop
       Calcular Distancia de Crowding (Frentes(i))
          NuevaPop = NuevaPop ∪ Frentes(i)
          i++
    Sorting por Distancia (Frentes(i))
    NuevaPop = NuevaPop ∪ Frentes(i)[1:(sizepop - |NuevaPop|)
    Hijos = Seleccion y Reproduccion(NuevaPop)
    generacion ++
    Poblacion(generacion) = NuevaPop
retornar Mejor Solucion Hallada
```

Figura 4: Esquema del algoritmo NSGA-II.

D. Algoritmos Evolutivos Paralelos

Las técnicas de paralelismo se aplican a los EAs para mejorar su eficiencia computacional y proporcionar un mecanismo diferente de exploración (Cantú-Paz, 2001). Dividiendo la población en elementos de procesamiento, los algoritmos evolutivos paralelos (pEAs) permiten acelerar la resolución de problemas que motivan el uso de poblaciones numerosas o múltiples evaluaciones de funciones objetivo con costo computacional elevado. Asimismo, los pEAs introducen un modelo de evolución diferente, que permite explotar la búsqueda simultánea sobre poblaciones distribuidas o estructuradas espacialmente.

La organización de la población constituye el principal criterio utilizado por los investigadores para clasificar los modelos de pEAs, destacándose tres grandes familias:

- pEAs maestro-esclavo, que distribuyen la evaluación de la función de fitness, mantienendo la evolución panmíctica característica de los modelos secuenciales.
- pEAs de población distribuida, que trabajan con subpoblaciones (islas) restringiendo las interacciones sólo entre individuos de la misma isla. Un operador de *migración* intercambia individuos entre islas, introduciendo una nueva fuente de diversidad.
- pEAs celulares, que poseen una estructura espacial subyacente a la población y un modelo de propagación de características de individuos (*difusión*), que sigue las direcciones definidas por la topología de conexión de elementos de procesamiento.

Estos modelos tienen diferentes variantes, y existen modelos *híbridos* que combinan las características de dos o más de los modelos de la categorización general presentada.

En la última década, los pEAs se han popularizado por su eficiencia computacional y aplicabilidad para la resolución de problemas en diversas áreas (Alba y Tomassini, 2002).

E. Algoritmos Evolutivos Paralelos para Optimización Multiobjetivo (pMOEAs)

Existen pocas propuestas de paralelismo aplicado a MOEAs, y en general su alcance es limitado (Coello et al., 2002). La referencia inmediata es el artículo de Veldhuizen et al. (2003) donde se exponen detalles de diseño, implementación y testeo de pMOEAs.

Al momento de escribir este artículo, el autor tiene conocimiento de cuatro antecedentes de aplicación de paralelismo a variantes del algoritmo NSGA. Dos de ellas proponen versiones paralelas modelo maestro-esclavo aplicadas a fluidodinámica, una tercera corresponde a un modelo de poblaciones distribuidas con división de dominio y la restante propone un modelo de islas aplicado a un problema de diseño de redes de comunicaciones confiables.

Mäkinen et al. (1996) propusieron un modelo maestro-esclavo del NSGA, implementado utilizando la biblioteca de pasaje de mensajes MPI, para evaluar eficientemente costosas funciones de fitness en un problema de diseño de perfiles aerodinámicos. Los autores sustituyeron la ruleta por el torneo como método de selección y modificaron el mecanismo de ***sharing*** por una variante denominada ***tournament slot sharing***, que fija al parámetro σ un valor relativo a un slot del torneo; e incorporaron elitismo, perpetuando los individuos no dominados en cada generación. Se reportan ejecuciones sobre un IBM SP2 con 8 procesadores modelo 390, presentando buenos valores de eficiencia, aunque el tiempo total de ejecución de una optimización resulta, aún para el modelo paralelo, excesivamente alto.

Al resolver otro problema de diseño aerodinámico, Marco et al. (1999) propusieron una estrategia de paralelismo en dos niveles para el NSGA, que distribuye el cálculo de flujos Eulerianos en la función de fitness. Los autores reportan tiempos razonables para la resolución del problema sobre un equipo SGI Origin 2000 con 8 procesadores R10000 a 195 Mhz, pero no realizan comparaciones de eficiencia contra modelos seriales.

La propuesta de ***dominancia guiada por distribución*** de Deb et al. (2002) plantea una búsqueda concurrente usando división de dominio en el NSGA-II. Los autores sugieren guiar la búsqueda a diferentes secciones del frente de Pareto transformando los objetivos mediante una suma ponderada que modifica el concepto de dominancia y permite que los procesos se enfoquen en diferentes regiones de búsqueda. Asimismo, se introduce un operador de migración para posibilitar la cooperación entre procesos. Se reportan buenos resultados en cuanto a cobertura del frente de Pareto, y muy buenos resultados respecto a la eficiencia, logrando ***speedup*** superlineal en el conjunto de problemas de prueba estudiados.

Recientemente, Duarte et al. (2003) propusieron un modelo de subpoblaciones sobre una variante del NSGA potenciada con elitismo, aplicado a un problema de diseño de redes de comunicaciones donde la complejidad computacional está dada por las simulaciones necesarias para estimar la confiabilidad de la red. Los autores confirman la ventaja de utilizar el paralelismo, aunque reportan que la versión paralela del NSGA no logró superar la calidad de resultados obtenidos por la versión paralela del algoritmo SPEA con elitismo.

Las técnicas de procesamiento paralelo han sido aplicadas a otros MOEAs, pero tomando en cuenta que en general los problemas teóricos abordados no requieren grandes esfuerzos computacionales, el área ha suscitado una atención limitada para los investigadores.

IV. UNA VERSIÓN PARALELA DEL ALGORITMO NSGA-II

La idea del trabajo consistió en aplicar un modelo de paralelismo al NSGA-II, tratando de simplificar el diseño del algoritmo paralelo. La única modificación significativa realizada sobre el mecanismo evolutivo de la versión serial del NSGA-II consiste en trabajar con subpoblaciones e introducir un operador de migración. Esta sección presenta los detalles de diseño e implementación de la versión paralela diseñada.

A. Modelo de paralelismo

Se propuso trabajar sobre el *modelo de islas* presentado en la Figura 5. `SeleccionMigracion` determina la política para seleccionar los individuos `Emigrantes` a intercambiar con otra isla durante el proceso de `Migración`, aplicado asincrónicamente de acuerdo a un grafo que conecta las islas mediante un anillo unidireccional. Los individuos `Inmigrantes` sustituyen a individuos de la población destino de acuerdo a una política de reemplazo determinada.

```
Inicializar(Poblacion(0))
generacion = 0
mientras (no CriterioParada) hacer
  Evaluar(Poblacion(generacion))
  Operador de diversidad(Poblacion(generacion))
  Asignar fitness(Poblacion(generacion))
  Padres = Seleccion(Poblacion(generacion))
  Hijos = Operadores de Reproduccion(Padres)
  NuevaPop = Reemplazar(Hijos, Poblacion(generacion))
  generacion ++
  Poblacion(generacion) = NuevaPop
  Si (CondicionMigracion)
    Emigrantes = SeleccionMigracion(Poblacion(generacion))
    Inmigrantes = Migracion(Emigrantes)
    Insertar(Inmigrantes, Poblacion(generacion))
retornar Mejor Solucion Hallada
```

Figura 5: Esquema de un Algoritmo Evolutivo Multiobjetivo Paralelo de Población Distribuida.

Cada isla evoluciona hasta cumplirse el criterio de parada y luego envía sus individuos no dominados a una isla distinguida. Esta isla receptora aumenta dinámicamente el tamaño de su población y aplica el mecanismo evolutivo durante un número reducido de generaciones extra, potenciando al algoritmo distribuido con una interacción panmíctica que mejora la convergencia –aumentando el número de puntos no dominados globales– y mejora la distribución, al calcular distancias de *crowding* sobre la población global. Se estudiaron las ventajas de utilizar la interacción panmíctica, concluyéndose que con un número reducido de generaciones se logran significativas mejoras en el número de puntos no dominados.

El modelo se desarrolló sobre la versión 1.12 de la implementación MPICH de la biblioteca de desarrollo de programas paralelos y distribuidos MPI (MPI Forum, 2003).

B. Operadores

El proceso de migración se diseñó siguiendo las sugerencias de Veldhuizen et al. (2003). Se adoptó una estrategia de selección elitista aleatoria de individuos no dominados para los emigrantes, intentando obtener una presión de selección relativamente alta. Se trata de permitir la evolución semi-independiente de las islas, evitando la convergencia hacia un mismo conjunto de puntos no dominados, permitiendo la cooperación mediante el intercambio ocasional de individuos bien adaptados para la resolución del problema. La política de reemplazo sigue también una estrategia elitista, sustituyendo individuos dominados de la población destino por los inmigrantes arribados. Tomando en cuenta que para problemas continuos simples las islas obtienen rápidamente un conjunto de puntos no dominados que abarca el total de la población, se implementó una estrategia de reemplazo complementaria que utiliza como criterio la diversidad, eliminando individuos con valores de distancia de *crowding* bajos, considerando que existirán otros individuos que representen su sección del espacio fenotípico. Las migraciones se ejecutan asincrónicamente, ya que las operaciones de envío y recepción de individuos no bloquean la ejecución del algoritmo, permitiendo la evolución continua en cada isla. De acuerdo a la capacidad de procesamiento, el intercambio puede realizarse en diferentes generaciones en las distintas islas.

V. PROBLEMAS DE PRUEBA Y MÉTRICAS UTILIZADAS

Esta sección presenta el conjunto de problemas de prueba y las métricas utilizadas para evaluar la calidad de las soluciones y la eficiencia de la versión paralela diseñada.

A. Problemas de Prueba

El conjunto de problemas de prueba intentó abarcar una amplia gama de características. En general se trabajó con problemas continuos de dos objetivos, aunque se incluyeron casos discretos y con más objetivos, con frentes de Pareto convexos y cóncavos, conectados y desconectados, con seudofrentes y con diferentes distribuciones de puntos y restricciones.

Se consideraron doce problemas bien conocidos en el área, propuestos para evaluar pMOEAs por Coello et al. (2002) y Deb (2001): los seis problemas ZDT (Zitzler, Deb y Thiele, 2000), que poseen frentes de Pareto conocidos y permiten evaluar la calidad de las soluciones obtenidas, el problema SCH2 (Schaffer, 1984), el problema KUR (Kursawe, 1990) y problemas con restricciones: el problema CEX de Deb (2001), el problema BNH (Binh y Korn, 1997) y los problemas VNT2 y VNT3 (Viennet et al., 1996).

B. Métricas de desempeño y eficiencia

Existen varias métricas para evaluar la calidad de los resultados de MOEAs (Coello et al, 2002). En este trabajo se utilizan cuatro métricas, tomando en cuenta los propósitos de los MOEAs, la convergencia al frente de Pareto y la distribución de puntos no dominados:

- El número de puntos no dominados diferentes que halla el algoritmo (lo notaremos *q*).
- Distancia generacional (GD): distancia promedio entre los puntos del frente de Pareto calculado por el algoritmo y el frente de Pareto real, presentada en la Ecuación 6.1.
- Spacing: evalúa la distribución de puntos no dominados en el frente de Pareto calculado por el algoritmo (Schott, 1995), presentada en la Ecuación 6.2.
- Spread: propuesta por Deb (2000), utiliza como información adicional la distancia a los "extremos" del frente de Pareto para tener una medida más precisa de la cobertura del frente. Su expresión se presenta en la Ecuación 6.3.

$GD = \frac{1}{q}\left(\sum_{i=1}^{q}(d_i^{FP})^M\right)^{1/M}$	$Spacing = \sqrt{\frac{1}{q-1}\sum_{i=1}^{q}(\bar{d}-d_i)^2}$	$Spread = \frac{\sum_{k=1}^{M} d_k^{\,e} + \sum_{i=1}^{q}\left\lvert \bar{d}-d_i\right\rvert}{\sum_{k=1}^{M} d_k^{\,e} + q.\bar{d}}$
Ecuación 6.1: distancia generacional.	*Ecuación 6.2: spacing*	*Ecuación 6.3 : spread.*

Figura 6: Definición de las métricas utilizadas para evaluar la calidad de soluciones.

En la Figura 6, d_i^{FP} es la distancia en el espacio de las funciones objetivo entre la solución ***i-ésima*** del frente de Pareto calculado y el punto más próximo del frente de Pareto real. d_i es la distancia entre la solución ***i-ésima*** y su vecino más próximo, la solución ***j-ésima***, en el frente de Pareto calculado: $d_i = \min_j(\left|f_1^i(x) - f_1^j(x)\right| + ... + \left|f_M^i(x) - f_M^j(x)\right|)$, $i = 1,...,q$, $\bar{d}$ es el promedio de los d_i. $d_k^{\,e}$ es la distancia entre el extremo del frente de Pareto real, tomando en cuenta la ***k-ésima*** función objetivo, y el punto más cercano del frente calculado.

Para evaluar la eficiencia computacional, se adopta el criterio habitual de determinar el ***speedup***, que mide la relación entre el tiempo medio de ejecución del algoritmo ejecutando sobre un procesador y el tiempo medio de ejecución utilizando *m* procesadores, y el valor de ***eficiencia***, medida que normaliza el valor del speedup.

VI. RESULTADOS

Esta sección presenta los experimentos comparativos realizados para evaluar la versión paralela, los resultados obtenidos y su análisis. Para cada problema considerado se realizaron 30 ejecuciones independientes de las versiones paralela y serial del NSGA-II, sobre un cluster de computadores Intel Pentium IV 2.4 GHz, cada uno con 512 Mb RAM y sistema operativo SuSE Linux 8.0, conectados por LAN Fast Ethernet a 100Mb/s.

A. Ajuste de operadores y parámetros

Tomando en cuenta el elevado número de experimentos necesarios, no se realizaron experimentos para hallar una configuración óptima de los parámetros del algoritmo serial para cada problema, ni se realizaron pruebas para determinar la influencia de los operadores cuando existen alternativas, utilizándose cruzamiento y mutación simple.

Los experimentos fueron realizados con la configuración de parámetros que se ofrece en la Figura 7. Los valores de probabilidad fueron tomados de Deb et al. (2000) quien los utiliza para la resolución de cinco problemas ZDT y problemas con restricciones, entre otros. Aún cuando la configuración paramétrica no sea la adecuada para obtener los mejores resultados para todos los problemas considerados, los valores usados influenciarán tanto al modelo secuencial como el paralelo, sin afectar de sobremanera el estudio comparativo.

Se usó un criterio de parada de esfuerzo prefijado, trabajando con un valor de tope alto (500 generaciones), tratando de dar al algoritmo mayor capacidad de alcanzar buenos resultados.

Inicialmente se propuso trabajar con poblaciones de 100 individuos. La escasa complejidad de las funciones objetivo en los problemas considerados, llevó a incrementar a 400 individuos para poder apreciar la mejora de eficiencia computacional del modelo paralelo.

Parámetro	***Valor***
Probabilidad de cruzamiento	0.9
Probabilidad de mutación	1/*N* siendo *N* el número de variables en codificación real o el largo de string en codificación binaria
Índice de distribución para cruzamiento real	20
Índice de distribución para mutación real	20
Criterio de parada	500 generaciones
Tamaño de la población	400 individuos
Parámetros Adicionales para la Versión Paralela	
Número de subpoblaciones utilizadas	4 subpoblaciones
Frecuencia de migración	25 generaciones
Individuos participantes en la migración	5 individuos
Topología de migración	Anillo unidireccional
Generaciones de interacción panmícitica	10 generaciones

Figura 7: Parámetros utilizados.

Se realizaron experimentos no formalizados sobre los problemas ZDT1 y ZDT2 para determinar valores adecuados para los parámetros del modelo de migración, observándose que la escasa complejidad de los problemas involucrados hacía a los resultados poco sensibles a variaciones en los parámetros. Se decidió utilizar un valor alto para la frecuencia de migración (25 generaciones) y un valor pequeño para el número de individuos intercambiados en cada migración (5 individuos). Disminuyendo la frecuencia de migración o incrementando el número de individuos participantes, la versión distribuida tendría un comportamiento similar al modelo serial, situación que se intentó evitar.

Asimismo, se investigó la influencia del número de generaciones de interacción panmíctica en los resultados, mostrando que un valor bajo de 10 generaciones era suficiente para lograr un número de puntos no dominados similar a la obtenida por el algoritmo serial.

B. Resultados

La Figura 8 resume los resultados de los modelos serial y paralelo del NSGA-II sobre los problemas considerados. Se presentan valores promedio y desviación estándar de tiempo de ejecución (segundos), puntos no dominados diferentes, distancia generacional, spread y spacing, obtenidos con una población de 400 individuos y un criterio de parada de 500 generaciones, valores que permiten apreciar las ventajas de eficiencia del algoritmo paralelo.

El cálculo de la distancia generacional asume conocido el frente de Pareto real para cada problema. En los casos de los problemas ZDT, SCH2, CEX y BNH donde existe una expresión analítica para el frente de Pareto, se generó el frente usando una discretización de paso 10^{-5}. Para el resto de los problemas, se utilizaron como valores ideales del frente de Pareto los calculados por Nebro et al. (2003) mediante una técnica enumerativa basada en evaluar los puntos del espacio discretizado. Para el problema ZDT5, con espacio de búsqueda discretizado en valores enteros de la función f_1 no se calcularon las métricas de diversidad. En este problema, el número de puntos no dominados diferentes es en sí mismo una medida de que tan bien se muestrean los 31 puntos del frente de Pareto discreto.

Prob.	SCH2				ZDT1				ZDT2				ZDT3			
Modelo	Serial		Paralelo		Serial		Paralelo		Serial		Paralelo		Serial		Paralelo	
Medida	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv
Tiempo (s.)	36.1	0.14	3.52	0.08	21.67	1.74	5.43	0.17	26	0.52	5.58	0.16	29.24	2.42	6.32	0.92
Puntos ND	379	4.44	370	4.54	368	5.4	366.5	6.2	366.5	5.49	365.5	5.29	368	4.6	370	4.2
GD	1.E-4	0	1.E-4	0	7.E-6	4.E-6	8.E-6	2.E-6	4.E-6	1.E-6	4.E-6	1.E-6	24.E-6	19.E-7	24.E-6	19.E-7
Spread	0.767	0.011	0.776	0.011	0.406	0.013	0.406	0.014	0.401	0.014	0.398	0.02	0.401	0.013	0.402	0.018
Spacing	0.137	0.002	0.138	0.002	0.0611	0.0007	0.0611	0.001	0.061	0.001	0.062	0.001	0.0623	0.001	0.0622	0.0008

Prob.	ZDT4				ZDT5				ZDT6				KUR			
Modelo	Serial		Paralelo		Serial		Paralelo		Serial		Paralelo		Serial		Paralelo	
Medida	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv
Tiempo (s.)	26.41	2.86	4.92	0.77	34.65	3.54	9.13	1.06	25.9	0.39	3.06	0.14	20.57	0.1	3.02	0.11
Puntos ND	390	3.69	390	3.69	25	0	25	1.75	361.5	5.84	362	6.23	387.5	3.73	388	2.97
GD	56.E-6	52.E-6	18.E-5	68.E-6	0.3445	0	0.2283	0.0724	0.0715	0.0065	0.07	0.0041	0.0094	0.0006	0.0096	0.0005
Spread	0.368	0.012	0.359	0.02	-	-	-	-	0.466	0.026	0.433	0.018	0.483	0.038	0.452	0.021
Spacing	0.061	0.001	0.061	0.001	-	-	-	-	0.11	0.003	0.056	0.001	0.322	0.008	0.163	0.002

Prob.	CEX				BNH				VI2				VI3			
Modelo	Serial		Paralelo		Serial		Paralelo		Serial		Paralelo		Serial		Paralelo	
Medida	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv	Prom	Desv
Tiempo (s.)	21.62	0.04	2.9	0.07	27.31	0.3	3.54	1.05	25.84	0.03	3.24	0.22	23.28	0.16	3.95	0.07
Puntos ND	355.5	5.52	352	6.41	395.5	2.85	395	1.97	391	2.72	392.5	3.2	374.5	4.47	375.5	4.96
GD	43.E-5	79.E-6	59.E-5	84E-6	0.0132	0.0011	0.0152	0.0013	0.0015	0.0002	0.0015	0.0002	0.0223	0.0006	0.0222	0.0004
Spread	0.734	0.015	0.942	0.027	0.45	0.015	0.445	0.014	0.6137	0.027	0.6296	0.03	0.7519	0.028	0.7638	0.029
Spacing	0.125	0.002	0.114	0.003	0.577	0.006	0.579	0.004	0.097	0.002	0.087	0.002	0.154	0.003	0.155	0.002

Figura 8: Resultados obtenidos sobre los problemas de prueba.

El algoritmo paralelo obtuvo resultados de similar calidad que el algoritmo serial, tanto al evaluar la aproximación al frente de Pareto como la diversidad de las soluciones finales. Para el problema ZDT5 el algoritmo paralelo alcanzó mejores valores de distancia generacional, mientras que para los problemas ZDT6 y KUR el algoritmo paralelo alcanzó mejores valores de distribución de puntos. Por otra parte, para el problema ZDT4 el algoritmo paralelo alcanza un peor valor de distancia generacional que el algoritmo serial.

La Figura 9 presenta resultados de experimentos con poblaciones de 100 individuos y 200 generaciones, para los problemas en que el modelo paralelo mejoró los resultados del serial. El algoritmo paralelo obtuvo una significativa mejora en la distancia generacional para el problema ZDT5, donde el algoritmo serial obtiene siempre el mismo conjunto de soluciones no dominadas, y por ello el mismo valor de distancia generacional. El algoritmo paralelo mejora la diversidad, alcanzando un número mayor de puntos no dominados, que le permite disminuir en un factor de 10 la distancia promedio al Frente de Pareto. En el problema ZDT6 ambas versiones del algoritmo obtienen el mismo número de puntos no dominados, pero la versión paralela muestrea casi de forma "perfecta" el frente de Pareto.

Problema	*Modelo*	*Medida*	*Tiempo (s.)*	*Puntos ND*	*GD*	*Spread*	*Spacing*
ZDT5	Serial	Promedio	2.08	23	0.5912	-	-
		Desv. Est.	0.04	0	0.0	-	-
	Paralelo	Promedio	2.51	30	0.0511	-	-
		Desv. Est.	0.21	1	0.011	-	-
ZDT6	Serial	Promedio	0.76	89	0.07418	0.466	0.110
		Desv. Est.	0.01	3.09	0.00463	0.026	0.003
	Paralelo	Promedio	1.12	89	0.00024	0.415	0.113
		Desv. Est.	0.03	2.41	0.02436	0.047	0.004
KUR	Serial	Promedio	2.25	98	0.0128	0.908	0.322
		Desv. Est.	0.24	0.97	0.0023	0.008	0.008
	Paralelo	Promedio	3.02	98	0.015	0.874	0.288
		Desv. Est.	0.11	1.34	0.003	0.011	0.020

Figura 9: Resultados para los problemas donde la versión paralela obtuvo mejores resultados que la serial.

En lo referente a la eficiencia computacional, la Figura 10 presenta los tiempos de ejecución (segundos) y valores de speedup y eficiencia obtenidos para cada problema.

		SCH2	ZDT1	ZDT2	ZDT3	ZDT4	ZDT5	ZDT6	KUR	CEX	BNH	VI2	VI3
Modelo	Tiempo Serial (s)	36.10	21.67	26.00	29.24	26.41	34.65	25.90	20.57	21.62	27.31	25.84	23.28
	Tiempo Paralelo (s)	3.52	5.43	5.58	6.32	4.92	9.13	3.06	3.02	2.90	3.54	3.24	3.95
Medida	Speedup	10.26	3.99	4.66	4.63	5.37	3.80	8.47	6.82	7.45	7.73	7.98	5.89
	Eficiencia	2.56	0.99	1.16	1.15	1.34	0.95	2.12	1.71	1.86	1.93	1.99	1.47

Figura 10: Evaluación del desempeño.

Puede apreciarse la notoria mejora de eficiencia del modelo paralelo utilizando 4 procesadores. Se obtuvo un speedup superlineal para todos los casos estudiados, salvo en los problemas ZDT1 y ZDT5 donde el comportamiento correspondió a un speedup lineal. Estos resultados confirman la argumentación de Deb et al. (2002), quienes reportan alcanzar speedup superlineal al aplicar paralelismo por división de dominio al NSGA-II, aunque no se presentan resultados numéricos de evaluación de eficiencia. En nuestro entorno de trabajo, este comportamiento ha sido ratificado al aplicar el modelo paralelo a un problema de diseño de redes de comunicaciones confiables (Nesmachnow, 2004).

VII. Conclusiones y Trabajo Futuro

Este trabajo ha presentado una versión paralela del algoritmo evolutivo para optimización multiobjetivo NSGA-II. La propuesta se basó en aplicar un paradigma de diseño sencillo para el algoritmo paralelo, modificando mínimamente la estructura del algoritmo. Se utilizó un modelo de islas con migración, para tomar ventajas de infraestructuras disponibles para resolver complejos problemas de optimización con altos requerimientos de cómputo.

La versión paralela del algoritmo se evaluó sobre doce problemas estándar, cotejando su calidad de resultados y su eficiencia computacional con los obtenidos por la versión serial.

El modelo paralelo mostró muy buenos resultados de eficiencia computacional, alcanzando valores de speedup superlineal al trabajar con poblaciones numerosas. Respecto a la calidad de resultados, solamente en tres problemas se detectaron mejoras significativas sobre los resultados del algoritmo serial, en especial al utilizar poblaciones de tamaño mediano y un número no elevado de generaciones como criterio de parada.

Las líneas de trabajo futuro comprenden el estudio de otras estrategias de paralelismo, que permitan abordar problemas más complejos dividiendo el espacio de búsqueda o mediante la evolución heterogénea de las islas, y el análisis de políticas de migración y reemplazo, profundizando el estudio de la influencia de la estrategia de centralización utilizada.

REFERENCIAS BIBLIOGRÁFICAS

Alba E., Tomassini M. (2002). *Parallelism and Genetic Algorithms*, IEEE Transactions on Evolutionary Computation 6(5), pp. 443-462.

Binh T., Korn U. (1997), *Multiobjective Evolution Strategy for Constrained Optimization Problems*, Proceedings of the IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, pp. 357-362.

Cantú-Paz E. (2001). *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers.

Coello C., Van Veldhuizen D., Lamont G. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems.* Kluwer Academic Publishers.

Deb K., Agrawal S., Pratab A., Meyarivan T. (2000). *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*, Proceedings of the Parallel Problem Solving from Nature VI Conference, pp. 849-858.

Deb K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, New York.

Deb K., Zope P., Jain A. (2002). *Distributed Computing of Pareto-Optimal Solutions Using Multi-Objective Evolutionary Algorithms*. Report No. 2002008, Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology Kanpur. Disponible en http://www.iitk.ac.in/kangal/pub.htm.

Duarte S., Barán B., Benítez D. (2003). *Telecommunication Network Design with Parallel Multiobjective Evolutionary Algorithms*. IFIP/ACM Latin America Networking Conference, pp 1-11.

Goldberg D. (1989). *Genetic Algorithms in Search,Optimization and Machine Learning.* Addison-Wesley.

Kursawe F. (1991). *A variant of evolution strategies for vector optimization*, Parallel Problem Solving from Nature, Lecture Notes in Computer Science vol. 496, pp. 193-197, Springer-Verlag

Mäkinen R., Neittaanmäki P., Periaux J., Sefrioui M., Toivanen J (1995). *Parallel genetic solution for multiobjective MDO*. Parallel CFD'96 Conference, pp. 352-359, Elsevier.

Marco N., Lanteri S., Desideri J.., Périaux J. (1999), *A Parallel Genetic Algorithm for Multi-Objective Optimization in Computational Fluid Dynamics*, Evolutionary Algorithms in Engineering and Computer Science, pp. 445-456. Wiley, Chichester, UK.

MPI Forum (2003), *MPI (Message Passing Interface*) *Forum Home Page*. Disponible en línea http://www.mpi-forum.org/. Consultada diciembre 2003.

Nebro A., Alba E., Luna F. (2003), *Optimización Multiobjetivo y Computación Grid*, Actas del Tercer Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, pp. 365-372.

Nesmachnow S. (2004), *Una Versión Paralela del Algoritmo Evolutivo para Optimización Multiobjetivo NSGA-II y su Aplicación al Diseño de Redes de Comunicaciones Confiables*. Reporte Técnico 04-03, Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Uruguay.

Schaffer D. (1984). *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms***.** PhD thesis, Vanderbilt University, USA.

Schott J. (1995). *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization*. Master's thesis, Massachusetts Institute of Technology, USA.

Srinivas N., Deb K. (1994), *Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms.* Evolutionary Computation 2(3), pp. 221-248.

Van Veldhuizen D., Zydallis J., Lamont G. (2003), *Considerations in engineering parallel multiobjective evolutionary algorithms*. Evolutionary Computation 7(2), pp. 144-173, IEEE Press.

Viennet, R., Fontiex, C., Marc, I. (1996), *Multicriteria Optimization Using a Genetic Algorithm for Determining a Pareto Set*, Journal of Systems Science 27(2), pp. 255-260.

Zitzler E., Deb K., Thiele L. (2000), *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results.* Evolutionary Computation 8(2), pp. 173-195.