Optimización de Redes Ópticas WDM Mediante Paralelismo e Inteligencia Colectiva

Álvaro Rubio Largo [1], Miguel A. Vega Rodríguez [1], Juan A. Gómez Pulido [1] y Juan M. Sánchez Pérez [1]

Resumen— **En este trabajo se presenta un algoritmo multiobjetivo paralelo basado en el comportamiento de las abejas recolectoras de miel para resolver el problema Traffic Grooming. Este problema de Telecomunicaciones es considerado un problema NP-completo, por lo que la utilización de metaheurísticas y paralelismo de forma conjunta para optimizar redes ópticas es muy apropiado para reducir los tiempos de ejecución. Este algoritmo multiobjectivo ha sido paralelizado utilizando directivas OpenMP. Con el fin de evaluar el rendimiento de la propuesta, se han realizado diversos experimentos con 2, 4, 8 y 16 núcleos. Para llevar a cabo estos experimentos se han utilizado dos topologías de red óptica. Finalmente, se presenta una comparativa entre la técnica propuesta y otros métodos publicados en la literatura. De esta comparativa concluimos que este algoritmo basado en el comportamiento de las abejas supera los resultados obtenidos por otras heurísticas publicadas anteriormente.**

Palabras clave— **Multi-núcleo, OpenMP, artificial bee colony, traffic grooming, optimización multiobjetivo.**

I. Introduction

En los últimos años, el tráfico de telecomunicaciones ha crecido de manera exponencial en las redes de área extensa, metropolitanas y locales. Sin embargo, las redes de datos actuales no están preparadas para soportar este elevado número de demandas, ya que el ancho de banda está restringido a los Gbps. De esta forma, todo apunta a que las futuras redes de datos serán ópticas, ya que ofrecen un ancho de banda en el rango de Tbps.

La técnica más prometedora para explotar el enorme ancho de banda de las redes ópticas está basada en la Multiplexación por División de Longitud de Onda (Wavelength Division Multiplexing, WDM). Esta tecnología multiplica la capacidad de cada enlace de fibra óptica utilizando cientos de canales paralelos [1], cada canal en una longitud de onda distinta (λ). Estos canales soportan demandas de tráfico en el rango de Gbps (por ejemplo OC-48, OC-192 y OC-768).

Desgraciadamente, la mayor parte de aplicaciones y dispositivos están limitados por su velocidad de procesamiento (unos pocos Mbps), lo que se traduce en un gasto de ancho de banda en cada canal. Para solventar este problema, se pueden multiplexar varias demandas de baja velocidad (Mbps) en un mismo canal o longitud de onda. La conexión establecida punto-a-punto entre un nodo origen y un nodo destino se conoce como "lightpath". Este problema se conoce como problema Traffic Grooming.

Según el patrón de tráfico, este problema puede ser considerado *estático* o *dinámico*. Por un lado, si las demandas y velocidades de las mismas son conocidas a priori, el problema se soluciona diseñando una topología virtual para establecer el mayor número de demandas de tráfico posibles. Por otro lado, si el modelo de tráfico es dinámico, la topología virtual necesita ser reconfigurada cada vez que una demanda de tráfico entra o sale de la cola de peticiones [2]. En este trabajo, nos centraremos en resolver este problema de telecomunicaciones siguiendo un modelo de tráfico estático.

Además, dependiendo del número de lightpaths utilizados por una demanda para ser establecida, podemos distinguir dos tipos: *single-hop traffic grooming*, en caso de que las demandas únicamente puedan atravesar un lightpath; y *multi-hop traffic grooming*, donde las demandas pueden utilizar varios lightpaths concatenados.

Dado que el problema Traffic Grooming es un problema NP completo que se hace computacionalmente intratable a medida que aumenta el número de nodos [3], el objetivo principal de este trabajo es abordar este problema típico de telecomunicaciones utilizando computación evolutiva multiobjetivo, paralelismo e inteligencia colectiva con el fin de resolverlo de manera eficiente en un tiempo razonable. El Algoritmo Evolutivo Multiobjetivo (Multiobjective Evolutionary Algorithm, MOEA) elegido es una variante multiobjetivo de un algoritmo inspirado el comportamiento de las abejas de la miel, el algoritmo Artificial Bee Colony (ABC) [4], nos referiremos a él como MOABC. Se han realizado varios experimentos para ajustar el algoritmo MOABC, al igual que hemos medido el factor de aceleración y la eficiencia de la versión paralela con 2, 4, 8 y 16 núcleos, con el fin de demostrar que el algoritmo MOABC es altamente paralelizable. Finalmente, se presenta una comparativa entre el algoritmo MOABC y diversas técnicas propuestas en la literatura.

El resto del artículo se organiza como sigue. Una definición formal del problema Traffic Grooming se presenta en la Sección II. En la Sección III se decribe la versión Multiobjetivo basada en el algoritmo Artificial Bee Colony (MOABC), así como la versión paralela del mismo. En la Sección IV se presentan todos los experimentos y comparativas entre la técnica propuesta y otros métodos publicados en la literatura. Por último, en la Sección V aparecen las conclu-

[1]Dep. Tecnología de Computadores y Comunicaciones, Universidad de Extremadura, Escuela Politécnica, Campus Universitario s/n, 10003 Cáceres, e-mail: {arl, mavega, jangomez, sanperez}@unex.es

siones y las líneas de trabajo futuro.

II. Traffic Grooming: Definición Formal

En este trabajo, una topología de red óptica se define como un grafo dirigido G=(N, E), donde N es el conjunto de nodos y E el conjunto de enlaces físicos que conectan dichos nodos. En primer lugar enumeraremos las consideraciones para modelar este problema.

- En todos los nodos, el número de enlaces de entrada será igual al número de enlaces de salida.
- Los nodos no estarán dotados de conmutadores de longitud de onda (λ), es decir, supondremos *Wavelength Continuity Constraint* [5].
- El conjunto de demandas de tráfico es conocido de antemano. Dado que la granularidad de las demandas es $x \in$\{1, 3, 12 y 48\} (donde el ancho de banda de un canal OC-x es $x \times 51{,}84Mb/s$), estas demandas no pueden ser dividas y enrutadas de forma separada.
- Supondremos *multi-hop grooming*, por lo que una demanda podrá ser enrutada a través de varios lightpaths.

En segundo lugar, detallaremos brevemente los diferentes parámetros:

- N: conjunto de nodos. Donde, $|N|$ es el número total de nodos de la red óptica.
- W: número de longitudes de onda (λ) que pueden ser multiplexados en un enlace de fibra óptica. Todos los enlaces soportan la multiplexación de W longitudes de onda.
- T_i, $\forall i \in N$: número de transmisores en el nodo i; $T_i \geq 1$.
- R_i, $\forall i \in N$: número de receptores en el nodo i; $R_i \geq 1$.
- K: número de rutas más cortas calculadas en función del número de saltos.
- P_{mn}, $\forall m, n \in N$: número de enlaces de fibra óptica que conectan los nodos m and n. Si $P_{mn} = P_{nm} = 0$, entonces no existe un enlace físico entre los nodos m and n, en caso contrario P_{mn}=P_{nm}=1.
- C: capacidad de cada canal o enlace de fibra óptica, es decir, OC-48 si $C = 48$.
- Λ: matriz de demandas de tráfico

$$\Lambda^x = [\ \Lambda^x_{sd};\ s, d \in N\ ]_{|N|\times|N|},$$

  where Λ^x_{sd} es el número de conexiones OC-x entre el par de nodos (s, d); $x \in$\{1, 3, 12 y 48\}.
- d_{mn}, $\forall m, n \in N$: retardo de propagación en el enlace de fibra óptica desde el nodo m hasta el nodo n. Supondremos $d_{mn} = 1$.
- V^w_{ij}, $\forall w \in \{1 \ldots W\}, \forall i, j \in N$: número de lightpaths establecidos entre el nodo i y el nodo j utilizando la longitud de onda w (Topología Virtual). Si V^w_{ij} >1, los lightpaths entre los nodos i y j en la longitud de onda w estarán establecidos a través de rutas distintas. Aclarar que, V_{ij} es el número de total de lightpaths establecidos entre los nodos i y j.
- $P^{ij,w}_{mn}$, $\forall w \in \{1 \ldots W\}, \forall i, j, m, n \in N$: número de lightpaths entre los nodos i y j enrutados a través del enlace de fibra (m, n) sobre el canal w.
- S^x_{sd}, $x \in \{1, 3, 12$ y $48\}, \forall s, d \in N$: número de demandas OC-x enrutadas correctamente entre el nodo origen s y el nodo destino d. Si la t-ésima demanda OC-x desde el nodo origen s hasta el nodo destino d ha sido enrutada satisfactoriamente, entonces $S^{x,t}_{sd}$=1; en cualquier otro caso $S^{x,t}_{sd}$=0.

Por lo tanto, dada una topología de red óptica, un número fijo de transmisores y receptores en cada nodo, un número fijo de longitudes de onda disponibles en cada enlace de fibra, una capacidad para cada longitud de onda y un conjunto de demandas con diferente granularidad, el problema Traffic Grooming puede ser formulado como un Problema de Optimización Multiobjetivo (MOOP) [6], donde las funciones objetivo son:

- *Throughput de la Red* (f_1): Maximizar el número total de demandas correctamente enrutadas en la topología virtual.

$$Max\ \ \sum_{s=1}^{|N|} \sum_{d=1}^{|N|} \sum_{t=1}^{\Lambda^x_{sd}} (x \times S^{x,t}_{sd}) \qquad x \in \{1, 3, 12\ y\ 48\} \tag{1}$$

- *Número de Transmisores/Receptores (T_i/R_i) o Ligthpaths* (f_2): Minimizar el número total de transmisores/receptores utilizados, es decir, el número de lightpaths establecidos.

$$Min\ \sum_{i=1}^{|N|} \sum_{j=1}^{|N|} \sum_{w=1}^{W} V^w_{ij} \tag{2}$$

- *Retardo de Propagación Medio* (Average Propagation Delay, APD, f_3). Minimizar el número medio de saltos de los lightpaths establecidos, ya que asumimos $d_{mn} = 1$ en todos los enlaces.

$$Min\ \frac{\sum_{i=1}^{|N|} \sum_{j=1}^{|N|} \sum_{m=1}^{|N|} \sum_{n=1}^{|N|} (d_{mn} \times \sum_{w=1}^{W} P^{ij,w}_{mn})}{\sum_{i=1}^{|N|} \sum_{j=1}^{|N|} \sum_{w=1}^{W} V^w_{ij}} \tag{3}$$

III. Paralelización del Algoritmo Multiobjective Artificial Bee Colony

Artificial Bee Colony (ABC) es un algoritmo evolutivo (Evolutionary Algorithm, EA) basado en población propuesto por D. Karaboga en 2005 [4]. El comportamiento de este EA está inspirado en el comportamiento de las abejas de la miel a la hora de descubrir y explorar fuentes de alimento.

En el algoritmo ABC, la población de individuos es definida como una colonia de abejas la cual está dividida en tres grupos: *abejas obreras* asociadas a una fuente específica de comida, *abejas observadoras* encargadas de explotar las fuentes de comida con mayor cantidad de néctar, y las *abejas exploradoras* que

tienen como finalidad explorar aleatoriamente el entorno para encontrar nuevas fuentes de comida. Por lo tanto, cada fuente de alimento es representada como una posible solución del problema, donde la cantidad de néctar de dicha fuente indicará la calidad de la solución.

Dado que en este trabajo estamos tratando un problema con múltiples objetivos, se propone una adaptación multiobjetivo del algoritmo ABC: MOABC. A continuación se presentan los pasos básicos de esta propuesta multiobjetivo:

1. Completar la primera mitad de la colonia C=$\{x_1 \dots x_{SN/2}\}$ con abejas obreras generadas de forma aleatoria, donde SN es el número total de abejas (tamaño de la población). Después, incializamos el conjunto de *Soluciones No-Dominadas* a vacío.
2. Por cada abeja obrera x_i, donde $i \in \{1 \dots SN/2\}$, se realiza una mutación (F) de la posición d del cromosoma con el objetivo de explotar el vecindario de dicha fuente de alimento. Para generar una abeja vecina (v_i), se aplica la ecuación (4), donde x_k es una abeja obrera elegida de manera aleatoria.

$$v_i^d = x_i^d + F\ (x_i^d - x_k^d) \tag{4}$$

   Tras la generación de v_i, se realiza una selección avariciosa entre ambas abejas, x_i y v_i con el objetivo de mantener la mejor abeja en C. Para poder decidir cuál de las dos abejas es mejor, se ha utilizado el concepto multiobjetivo de *Dominancia* ($\succ$) [6]. En caso de que la abeja x_i no sea reemplazada por v_i, se incrementa el valor *límite* de estancamiento asociado a x_i, de lo contrario el contador se inicializa a cero.
3. Dado que las abejas observadoras eligen de manera probabilística la fuente de alimento dependiendo de la información indicada por las abejas obreras, se debe crear un vector con las probabilidades que tiene cada abeja obrera de ser elegida, ver la ecuación (5).

$$P_i = \frac{MOFitness(x_i)}{\sum_{m=1}^{SN/2} MOFitness(x_m)} \tag{5}$$

   Como podemos ver en la ecuación (5), para calcular la probabilidad que una abeja obrera tiene de ser elegida, se debe calcular su valor de fitness (*MOFitness*), ver la ecuación (6).

$$MOFitness(x_i) = (2^{x_i.r} + \tfrac{1}{1+x_i.cd})^{-1} \tag{6}$$

   Donde r es el valor de *rank* y cd es el valor de *crowding distance*, ambos conceptos son bien-conocidos en el campo multiobjetivo, para más información, consultar [7]. *MOFitness* debe ser máximo, por lo que, si comparamos el peor individuo con mejor valor de *rank* (*rank* 1) y menor valor de *crowding distance* (≈ 0); con el mejor individuo con *rank* 2 y valor más alto de *crowding distance* (∞), el valor de *MOFitness* será 1/3 y 1/4 respectivamente.
4. Por cada abeja observadora x_j, donde $j \in \{(SN/2+1) \dots SN\}$, elegimos de manera probabilística una abeja obrera (x_i) haciendo uso del vector de probabilidades calculado en el paso 3. Una vez elegida, la abeja observadora se coloca en la fuente de alimento asociada a la abeja x_i. Después, de igual forma que en el paso 2, la observadora generará una abeja vecina (v_j) haciendo uso de la ecuación (4). Por último, se selecciona de forma avariciosa entre las dos abejas (x_i y v_j) con el fin de mantener la mejor en la colonia C. En este caso, en la selección avariciosa se aplica el operador multiobjetivo $\succeq$ en lugar de $\succ$ para promover la diversidad de fuentes de alimento en la colonia.
5. Se comprueba para cada abeja si el valor *límite* de estancamiento ha alcanzado el valor máximo permitido (*max_limit*). En ese caso, dicha abeja se convertirá en abeja exploradora. Este tipo de abejas explorarán el espacio de búsqueda con el fin de descubrir nuevas fuentes de comida.
6. Se calcula el valor *MOFitness* de cada abeja según la equation (6) y se procede a la ordenación de la colonia en base a la calidad de las soluciones. De esta forma, en la primera mitad de la colonia se encontrarán aquellas abejas asociadas a las fuentes de alimento con más nectar (mayor calidad). Además, actualizamos el conjunto de *Soluciones No-Dominadas*.
7. Devolver el conjunto de *Soluciones No-Dominadas* si se satisface el criterio de parada, en caso contrario, volver al paso 2.

En este trabajo, se presenta una versión paralela del algoritmo MOABC utilizando librerias OpenMP. Las principales ventajas de OpenMP son que nos permiten explotar sistemas multi-núcleo con mínimas modificaciones del diseño original del algoritmo MOABC, y además no existen tiempos de comunicación entre procesos (memoria compartida), como ocurre con otras tecnologías como Message Passing Interface (MPI).

En el diseño del algoritmo paralelo MOABC, podemos distinguir cuatro regiones: *Región de Inicialización*, *Región de Abejas Obreras*, *Región de Abejas Observadoras* y *Región de Abejas Exploradoras*. Estas regiones son perfectamente paralelizables ya que no existen dependencias entre las tareas llevadas a cabo por los hilos en la misma región. En la Figura 1 se presenta el diseño del algoritmo paralelo MOABC con dos hilos (thr=2), y suponiendo un tamaño de colonia de 12 abejas (SN=12).

Como podemos ver en la Figura 1, en la *Región de Inicialización*, cada hilo se encarga de generar aleatoriamente $(SN/2)/thr$ abejas en la primera mitad de la colonia. Después, en la *Región de las Abejas Obreras*, las fuentes de alimento se explotan también de forma paralela. En la *Región de las Abejas Observadoras*, se asigna a cada abeja obrera un valor de *MOFitness* de forma paralela, pero un único hilo se encarga de calcular el vector de probabilidades. Finalmente, en la *Región de las Abejas Explorado-*

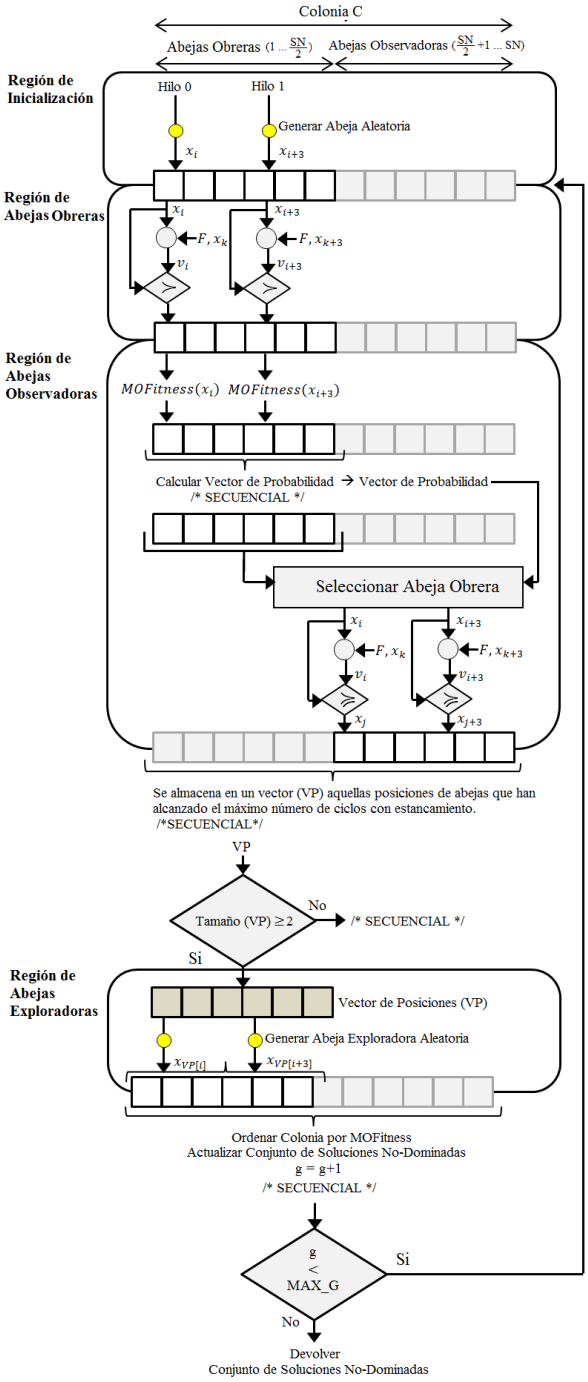


Fig. 1. Esquema del Algoritmo Paralelo MOABC

ras, se ha incorporado un vector que contendrá las posiciones de aquellas abejas que han alcanzado un número máximo de ciclos con estancamiento (*max_limit*). Si el número de abejas que han superado el umbral es mayor o igual a 2, se generan las abejas exploradoras de forma paralela, en caso contrario se encargará un único hilo. El proceso de ordenación y actualización se lleva a cabo de manera secuencial.

En conclusión, existen dependencias entre las distintas regiones, sin embargo, la mayor carga computacional ha sido paralelizada dentro de cada región con el fin de incrementar la eficiencia del algoritmo.

IV. Resultados Experimentales

En esta sección se presenta un conjunto de experimentos con el objetivo de probar la eficiencia de la versión paralela MOABC propuesta.

Con el fin de poder realizar comparativas con otras técnicas publicadas en la literatura, se han utilizado dos topologías ópticas diferentes. La primera es una pequeña topología con tan sólo seis nodos (6-node), una capacidad (C) en cada enlace de OC-48, y una matriz de tráfico con un total de 988 unidades OC-1. La segunda es una topología real, la red National Science Foundation (NSF), la cual consta de 14 nodos, una capacidad OC-192 y una matriz de tráfico de 5724 unidades OC-1. El algoritmo se ha probado sobre estas topologías variando el número de transmisores por nodo (T) y el número de longitudes de onda por enlace (W):

- Instancias 6-node: T={3,4,5,7} W={3} y T={3,4,5} W={4}. Un total de siete instancias. Nos referiremos a ellas como 6n-Tx-Wy; p.e. T=3,W=3 es 6n-T3-W3.
- Instancias NSF: T={3,4,5} W=3 y T={4,5,6} W={4}. Un total de seis instancias. Nos referiremos a ellas como NSF-Tx-Wy; p.e. T=3,W=3 es NSF-T3-W3.

Para más información acerca de las topologías, las matrices de tráfico o las instancias, consultar [3].

Tras realizar un ajuste de parámetros del algoritmo MOABC, la mejor configuración encontrada es: un tamaño de colonia (SN) de 100 individuos, un factor de mutación (F) del 25 % y un valor máximo de ciclos de estancamiento (*max_limit*) de 5 ciclos o generaciones. El máximo número de generaciones (MAX_G) será de 3000 y 7500 para las topologías 6-node y NSF, respectivamente.

Dado que la salida del algoritmo MOABC es un conjunto de *Soluciones No-Dominadas* (frente de Pareto), es necesario utilizar una métrica que nos indique la calidad del conjunto de soluciones. En este trabajo, se ha utilizado la conocida métrica de Hipervolumen (HV) [8], la cuál mide el hipervolumen entre el frente de salida y un punto de referencia (r). El punto de referencia depende del máximo valor de las tres funciones objetivo: máximo throughput (f_1), $T^*|N|$ (f_2) y $|N|$ (f_3). Por ejemplo, el punto de referencia de la instancia 6n-T4-W3 es r=(988, 24, 6).

La arquitectura utilizada para realizar estos experimentos es una máquina con 4-procesadores AMD Opteron Magny-Cours 6174 de 12 núcleos a 2.2GHz y 12MB Cache (Scientific Linux 6.1). El algoritmo paralelo MOABC fue compilado utilizando GCC 4.4.5. En los experimentos se presenta la mediana de 30 ejecuciones independientes con 1, 2, 4, 8 y 16 núcleos sobre cada topología y cada instancia.

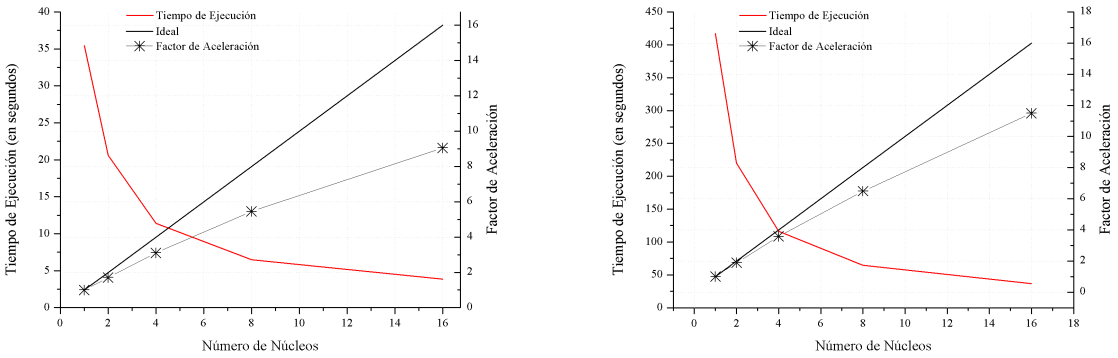
En primer lugar, comenzaremos con la topología 6-node. Como podemos observar en la Tabla I, el algoritmo MOABC ejecuta 3000 generaciones en 35,46s; sin embargo, podemos ver que si utilizamos 16 núcleos, es capaz de ejecutar el mismo número de generaciones hasta 9 veces más rápido (3,89s).

TABLA I

RESULTADOS EXPERIMENTALES (MEDIANA DE 30 EJECUCIONES)

	1 núcleo				2 núcleos				4 núcleos				8 núcleos				16 núcleos			
6-node	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E
6n-T3-W3*	$46{,}42\%_{0{,}01}$	34,10	1	100 %	$46{,}47\%_{0{,}03}$	19,87	1,72	85,79 %	$46{,}73\%_{0{,}03}$	10,97	3,11	77,68 %	$46{,}31\%_{0{,}04}$	6,26	5,45	68,13 %	$45{,}26\%_{0{,}01}$	3,74	9,05	56,56 %
6n-T4-W3*	$54{,}75\%_{0{,}05}$	35,86	1	100 %	$54{,}03\%_{0{,}01}$	20,43	1,76	87,76 %	$53{,}81\%_{0{,}01}$	11,31	3,17	79,23 %	$53{,}67\%_{0{,}03}$	6,46	5,55	69,37 %	$54{,}44\%_{0{,}02}$	3,86	9,24	57,74 %
6n-T5-W3*	$59{,}54\%_{0{,}05}$	34,42	1	100 %	$59{,}70\%_{0{,}05}$	21,02	1,64	81,86 %	$60{,}23\%_{0{,}05}$	11,55	2,98	74,48 %	$59{,}91\%_{0{,}01}$	6,58	5,23	65,37 %	$59{,}49\%_{0{,}01}$	3,97	8,62	53,89 %
6n-T7-W3*	$67{,}40\%_{0{,}05}$	39,39	1	100 %	$66{,}83\%_{0{,}04}$	21,77	1,81	90,49 %	$67{,}71\%_{0{,}05}$	12,04	3,27	81,81 %	$67{,}58\%_{0{,}04}$	6,80	5,79	72,37 %	$66{,}43\%_{0{,}04}$	4,09	9,57	59,83 %
6n-T3-W4*	$45{,}86\%_{0{,}03}$	33,29	1	100 %	$46{,}48\%_{0{,}05}$	19,88	1,67	83,72 %	$45{,}34\%_{0{,}05}$	10,99	3,03	75,70 %	$46{,}50\%_{0{,}05}$	6,37	5,23	65,34 %	$45{,}98\%_{0{,}04}$	3,72	8,89	55,58 %
6n-T4-W4*	$53{,}90\%_{0{,}01}$	34,79	1	100 %	$54{,}73\%_{0{,}05}$	20,36	1,71	85,45 %	$53{,}92\%_{0{,}05}$	11,27	3,09	77,17 %	$53{,}83\%_{0{,}04}$	6,51	5,35	66,84 %	$54{,}80\%_{0{,}05}$	3,87	8,93	55,81 %
6n-T5-W4*	$60{,}10\%_{0{,}04}$	36,35	1	100 %	$59{,}82\%_{0{,}04}$	20,77	1,75	87,51 %	$60{,}17\%_{0{,}03}$	11,58	3,14	78,44 %	$60{,}38\%_{0{,}03}$	6,58	5,53	69,10 %	$60{,}39\%_{0{,}01}$	3,97	9,08	56,77 %
$\overline{X}_{6n}$	$55{,}42\%_{0{,}03}$	35,46	1	100 %	$55{,}44\%_{0{,}04}$	20,59	1,72	86,08 %	$55{,}42\%_{0{,}04}$	11,39	3,11	77,79 %	$55{,}45\%_{0{,}04}$	6,51	5,45	68,07 %	$55{,}26\%_{0{,}03}$	3,89	9,06	56,60 %
	1 núcleo				2 núcleos				4 núcleos				8 núcleos				16 núcleos			
NSF	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E	HV_{IQR}	T	S	E
NSF-T3-W3*	$49{,}17\%_{0{,}00}$	381,37	1	100 %	$48{,}50\%_{0{,}07}$	205,37	1,86	92,85 %	$48{,}41\%_{0{,}06}$	104,80	3,64	90,98 %	$48{,}34\%_{0{,}07}$	58,63	6,50	81,31 %	$48{,}76\%_{0{,}07}$	32,17	11,82	73,89 %
NSF-T4-W3*	$57{,}76\%_{0{,}01}$	413,77	1	100 %	$58{,}44\%_{0{,}06}$	216,54	1,91	95,54 %	$57{,}78\%_{0{,}08}$	115,88	3,57	89,26 %	$58{,}80\%_{0{,}07}$	62,54	6,62	82,69 %	$57{,}58\%_{0{,}08}$	35,54	11,61	72,58 %
NSF-T5-W3*	$64{,}33\%_{0{,}05}$	460,02	1	100 %	$65{,}04\%_{0{,}07}$	240,37	1,91	95,69 %	$65{,}51\%_{0{,}07}$	125,14	3,68	91,90 %	$64{,}64\%_{0{,}07}$	68,62	6,70	83,80 %	$64{,}88\%_{0{,}05}$	39,09	11,75	73,46 %
NSF-T4-W4*	$57{,}64\%_{0{,}01}$	391,15	1	100 %	$58{,}42\%_{0{,}09}$	207,82	1,88	94,11 %	$58{,}17\%_{0{,}09}$	109,47	3,57	89,33 %	$57{,}45\%_{0{,}09}$	61,32	6,38	79,73 %	$58{,}60\%_{0{,}01}$	34,62	11,28	70,51 %
NSF-T5-W4*	$65{,}04\%_{0{,}01}$	401,33	1	100 %	$65{,}26\%_{0{,}07}$	219,43	1,83	91,45 %	$65{,}28\%_{0{,}07}$	115,26	3,48	87,05 %	$64{,}82\%_{0{,}07}$	64,77	6,20	77,46 %	$65{,}31\%_{0{,}02}$	37,11	10,79	67,45 %
NSF-T6-W4*	$70{,}16\%_{0{,}06}$	457,00	1	100 %	$69{,}83\%_{0{,}05}$	233,19	1,96	97,99 %	$69{,}77\%_{0{,}06}$	127,69	3,58	89,48 %	$69{,}30\%_{0{,}06}$	70,49	6,48	81,04 %	$69{,}77\%_{0{,}01}$	40,11	11,37	71,08 %
$\overline{X}_{NSF}$	$60{,}69\%_{0{,}02}$	417,44	1	100 %	$60{,}91\%_{0{,}07}$	220,46	1,89	94,60 %	$60{,}82\%_{0{,}07}$	116,37	3,59	89,67 %	$60{,}56\%_{0{,}07}$	64,39	6,48	81,01 %	$60{,}82\%_{0{,}04}$	36,44	11,44	71,49 %

HV: Hipervolumen mediana (Q2)
IQR: Rango intercuartílico
T: Tiempo de Ejecución mediana en segundos (Q2)
S: Factor de Aceleración mediana (Q2)
E: Eficiencia mediana (Q2)
* Indica que las diferencias de HV con 1, 2, 4, 8 y 16 núcleos no son estadísticamente significativas



(a) Topología 6-node

(b) Topología NSF

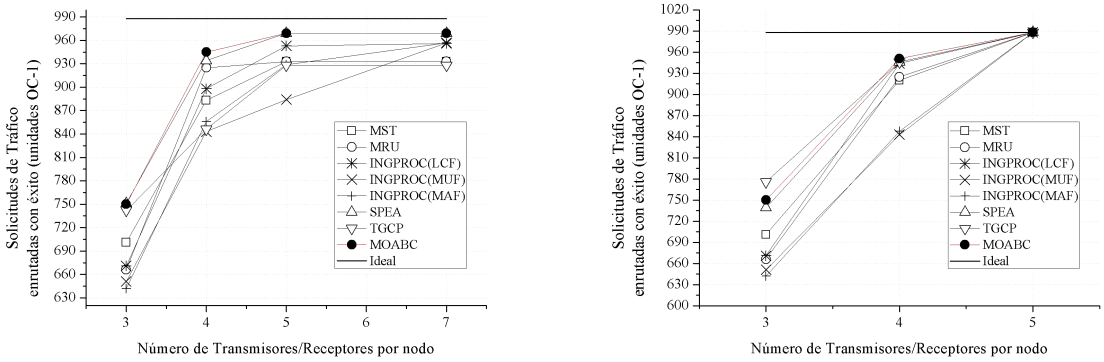
Fig. 2. Factor de Aceleración y Tiempo de Ejecución medio obtenido por el algoritmo paralelo MOABC

En segundo lugar, para la topología NSF, podemos ver que el algoritmo necesita mucho más tiempo para realizar 7500 generaciones, un tiempo medio de 417.44s. Se puede observar en la Tabla I, que esta topología es mucho más compleja ya que el total de demandas a establecer es mucho mayor que en la topología 6-node. En este caso, cuando el número de núcleos es 16, el algoritmo MOABC es capaz de ejecutar las 7500 generaciones en tan solo 36,44s, es decir, 11,44 veces más rápido.

En las figuras 2(a) y 2(b), presentamos la evolución de los tiempos de ejecución del algoritmo utilizando distinto número de núcleos. Además, se presenta el factor de aceleración medio. En resumen, podemos observar como la propuesta evolutiva basada en el comportamiento de las abejas obtiene eficiencias más altas a medida que el número de nodos de la red crece, en nuestro caso, con 16 núcleos, existe una diferencia de eficiencia del 15 %.

Con el fin de probar que la calidad de las soluciones obtenidas (HV) por el algoritmo MOABC no sufren cambios al aumentar el número de nodos, se ha llevado a cabo un análisis estadístico [9] de los resultados para determinar que las diferencias de HV obtenidas en las 30 ejecuciones independientes con 1, 2, 4, 8 y 16 núcleos no son estadísticamente significativas (p-value = 0,05). En la Tabla I se muestran los valores de HV obtenidos por el algoritmo MOABC con diferentes número de núcleos. Después de realizar el análisis estadístico, podemos concluir que las diferencias no son estadísticamente significativas, por lo que podemos afirmar que la calidad de las soluciones no se ven afectadas al aumentar el número de núcleos.

Por último, se presenta una comparativa entre el algoritmo MOABC y varias heurísticas que han abordado el problema Traffic Grooming en la literatura. Con el fin de realizar una comparativa justa, presentamos una comparativa visual del máximo valor (f_1) obtenido por cada heurística utilizando distinto número de longitudes de onda por enlace (W) y transmisores por nodo (T) para la topología 6-node. Las heurísticas con las que se realizarán las comparativas son: Maximizing Single-Hop Traffic (MST) [3], Maximizing Resource Utilization (MRU) [3], Strength Pareto Evolutionary Algorithm (SPEA) [10], Traffic Grooming based on Clique Partitioning (TGCP) [2] e INtegrated Grooming PROCedure (INGPROC) [11] con los esquemas de selección: *Least Cost First* (LCF), *Maximum Utilization First* (MUF) y *Maximum Amount First* (MAF);



(a) Topología 6-node, W=3

(b) Topología 6-node, W=4

Fig. 3. Comparativa entre el algoritmo MOABC y otros métodos publicados en la literatura

Si nos centramos en la Figura 3(a) y la Figura 3(b), podemos ver como el algoritmo MOABC obtiene unos valores de throughput en la mayoría de los casos superiores a los conseguidos por otras heurísticas publicadas en la literatura.

Tras realizar una serie se experimentos y una comparativa con otros métodos publicados en la literatura, se puede concluir, en primer lugar que el algoritmo MOABC es capaz de obtener una eficiencia media del 80 % con 2, 4, 8 y 16 núcleos; y en segundo lugar, que los resultados obtenidos son muy prometedores.

V. Conclusiones y Trabajo Futuro

En este trabajo se presenta una versión paralela de un Algoritmo Evolutivo Multiobjetivo basado en el algoritmo Artificial Bee Colony (MOABC) para resolver el problema Traffic Groomin en redes ópticas WDM. Utilizando el método propuesto se optimiza simultáneamente el throughput de la red, el número de transmisores utilizados, y el retardo de propagación medio.

En la fase experimental se ha medido el factor de aceleración y la eficiencia del algoritmo paralelo MOABC utilizando diferente número de núcleos (1, 2, 4, 8 y 16), dos topologías de red y 13 instancias; obteniendo una eficiencia media del 80 %. Además se han realizado comparativas entre el algoritmo MOABC y otros métodos publicados en la literatura. En conclusión, podemos decir que la propuesta paralela MOABC no sólo obtiene resultados de mayor calidad que otras heurísticas publicadas anteriormente, sino que es más eficiente.

Como línea de trabajo futuro se pretende aplicar otros algoritmos evolutivos al problema Traffic Grooming, así como el diseño de una versión paralela del algoritmo MOABC utilizando un modelo híbrido OpenMP + MPI para explotar varios sistemas multi-núcleo interconectados a través de una red de datos.

Agradecimentos

Álvaro Rubio-Largo es becario de investigación (PRE09010) del Gobierno de Extremadura (Consejería de Economía, Comercio e Innovación) y el Fondo Social Europeo (FSE). Por último, agradecer al Gobierno de Extremadura la ayuda GR10025 concedida al grupo TIC015.

Referencias

[1] K. Zhu and B. Mukherjee, "A Review of Traffic Grooming in WDM Optical Networks: Architectures and Challenges," *Optical Networks Magazine*, vol. 4, no. 2, pp. 55–64, 2003.

[2] Tanmay De, Ajit Pal, and Indranil Sengupta, "Traffic Grooming, Routing, and Wavelength Assignment in an Optical WDM Mesh Networks Based on Clique Partitioning," *Photonic Network Communications*, vol. 20, pp. 101–112, 2010.

[3] Keyao Zhu and B. Mukherjee, "Traffic Grooming in an Optical WDM Mesh Network," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 1, pp. 122–33, Jan. 2002.

[4] D. Karaboga, "An Idea Based on Honey Bee Swarm for Numerical Optimization," in *Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department*, 2005.

[5] Maurice Gagnaire, Mohamed Koubaa, and Nicolas Puech, "Network Dimensioning under Scheduled and Random Lightpath Demands in All-Optical WDM Networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. S-9, pp. 58–67, 2007.

[6] Kalyanmoy Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Inc., New York, NY, USA, 2001.

[7] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan, "A Fast Elitist Multi-Objective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, 2000.

[8] Eckart Zitzler and Lothar Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[9] David J. Sheskin, "Handbook of Parametric and Nonparametric Statistical Procedures," *4th ed. New York: Chapman & Hall/CRC Press*, 2007.

[10] Passakon Prathombutr, Jerry Stach, and E. K. Park, "An Algorithm for Traffic Grooming in WDM Optical Mesh Networks with Multiple Objectives," *Telecommunication Systems*, vol. 28, pp. 369–386, 2005.

[11] Hongyue Zhu, Hui Zang, Keyao Zhu, and Biswanath Mukherjee, "A Novel Generic Graph Model for Traffic Grooming in Heterogeneous WDM Mesh Networks," *IEEE/ACM Transaction on Networking*, vol. 11, pp. 285–299, 2003.