Algoritmos de Optimización multi-objetivos basados en colonias de hormigas

Julio Manuel Paciello Coronel
Universidad Nacional de Asunción, Facultad Politécnica
San Lorenzo, Paraguay, Casilla de Correos 1439
juliopaciello@gmail.com

Héctor Daniel Martínez Santacruz
Universidad Nacional de Asunción, Facultad Politécnica
San Lorenzo, Paraguay, Casilla de Correos 1439
hmartinez.py@gmail.com

Christian Gerardo Lezcano Ríos
Universidad Nacional de Asunción, Centro Nacional de Computación
San Lorenzo, Paraguay, Casilla de Correos 1439
clezcano@cnc.una.py

Benjamín Barán Cegla
Universidad Nacional de Asunción, Centro Nacional de Computación
San Lorenzo, Paraguay, Casilla de Correos 1439
bbaran@cnc.una.py

Abstract

Theory of ACO meta-heuristic is successfully applied to the resolution of combinatorial optimization problems. This paper presents a comparison using a benchmark of three bi-objective problems, QAP, TSP, and VRPTW, of the existing state-of-art algorithms in the resolution of multi-objective problems using ant colony theory. This paper proposes a new approach of multi-objective algorithm, the *Multiobjective Ant System*, proving its good behavior. Experimental results proves that the strategy of having only one table of pheromones and multiple visibilities empirically outperforms other strategies.

Keywords: Artificial Intelligence, Multi-objective optimization, Ant colonies, ACO meta-heuristic.

Resumen

La teoría de la meta-heurística ACO es utilizada con éxito en la resolución de problemas de optimización combinatoria. Este trabajo realiza una comparación utilizando tres problemas de prueba bi-objetivos, el QAP, TSP y el VRPTW, de diversos algoritmos ACO existentes en la actualidad que constituyen el estado del arte en la resolución de problemas multi-objetivos utilizando la teoría basada en colonias de hormigas. Se propone un nuevo algoritmo multi-objetivo, el *Multiobjective Ant System*, y se verifica un buen comportamiento empírico. Se demuestra empíricamente que la estrategia de utilizar una única tabla de feromonas y múltiples visibilidades supera a otras propuestas.

Palabras claves: Inteligencia Artificial, Optimización multi-objetivo, colonias de hormigas, meta-heurística ACO.

1. Introducción

Este trabajo presenta una comparación entre diversos algoritmos que utilizan la optimización multi-objetivo basada en el modelo del comportamiento de las colonias de hormigas reales [7, 20] y denominada metaheurística ACO (*Ant Colony Optimization*, u optimización basada en colonia de hormigas). El enfoque utilizado está orientado a comparar experimentalmente los algoritmos que constituyen el estado del arte en la optimización multi-objetivo basada en colonia de hormigas, en un conjunto de problemas de prueba bi-objetivos.

El trabajo puede ser considerado como una extensión del trabajo realizado por García-Martínez et al. en [16], considerando inclusive algoritmos propuestos recientemente como el M-MMAS [26], el MOA [18] y una nueva propuesta de este trabajo, el MAS *(Multiobjective Ant System)*. Además, se realizaron pruebas en un conjunto mayor de problemas. Se utilizaron reconocidos problemas de prueba de optimización multi-objetivo, el *Quadratic Assignment Problem* (QAP), definido en [4] y su versión multi-objetiva definida en [22], el *Traveling Salesman Problem* (TSP) [19] y el *Vehicle Routing Problem with Time Windows* (VRPTW) [2]. Estos problemas son considerados clásicos en la literatura de optimización combinatoria y del tipo NP-completos [27].

El trabajo está organizado como sigue: en la sección 2 se trata la formulación matemática de la optimización multi-objetivo, en la sección 3 se describen los algoritmos ACO multi-objetivos utilizados, y el nuevo algoritmo propuesto (MAS) es propuesto en la sección 4. Los resultados experimentales de la comparación se muestran en la sección 5, y finalmente en la sección 6 se presentan algunas conclusiones y trabajos futuros.

2. Optimización Multi-objetivo

La optimización multi-objetivo puede ser definida como el problema de encontrar un vector de variables de decisión que satisfacen restricciones y optimiza un vector de funciones cuyos elementos representan las funciones objetivo. Estas definiciones aparecen en los trabajos de Coello [5] y Deb [6].
Se desea encontrar un vector de decisión

$$\vec{x}=[x_{1,}x_{2,}\ldots,x_n]^T \tag{1}$$

con $\vec{x}\in\mathbb{R}^n$, que deberá satisfacer ω restricciones de desigualdad

$$g_i(\vec{x})\geq 0 \qquad i=1,2,\ldots,\omega \tag{2}$$

y optimizar el vector de funciones

$$\vec{f}(\vec{x})=[f_1(\vec{x}),f_2(\vec{x}),\ldots,f_b(\vec{x})]^T \tag{3}$$

que generalmente cumple con $\vec{f}(\vec{x})\in\mathbb{R}^b$. El conjunto de todas las soluciones que atienden (2) es conocido como dominio de soluciones factibles, y se representa como Ω , en general $\Omega\subset\mathbb{R}^n$.

El correspondiente conjunto imagen Ω_o se define como:

$$\Omega_o=\{\vec{f}(\vec{x})\in\mathbb{R}^b \mid \vec{x}\in\Omega\} \tag{4}$$

Definición 1: *Dominancia de Pareto:* Sean dos soluciones $u,v\in\Omega$. Se dice que *u* domina a *v* (denotado como $u \succ v$) si es mejor o igual que *v* en cada uno de los objetivos y estrictamente mejor en al menos un objetivo.
Como ejemplo, en un contexto de minimización $u \succ v$ si y solo si:

$$f_i(u)\leq f_i(v)\forall i\in[1,2,\ldots,b] \ \wedge \ \exists j\in[1,2,\ldots,b] \mid f_j(u)<f_j(v) \tag{5}$$

Definición 2: *Soluciones no comparables:* Dados $u,v\in\Omega$, si $u \nsucc v$ ni $v \nsucc u$, se dice que son soluciones no comparables, lo que se denota como $u \sim v$.
Definición 3: *Conjunto Pareto:* El conjunto de todas las soluciones $\vec{x}$ no dominadas en Ω se denomina Conjunto Pareto, lo que se denota como *CP*. Las soluciones $\vec{x}$ que pertenecen a *CP* se denotarán como x^*.
Definición 4: La imagen del Conjunto Pareto a través de la función $\vec{f}$ se denomina Frente Pareto, denotado por *Y*.

3. Optimización basada en colonia de hormigas

Un algoritmo ACO constituye un conjunto de agentes computacionales concurrentes (la colonia de hormigas artificiales), que se mueven a través de estados del problema correspondientes a soluciones parciales del mismo, buscando las mejores soluciones factibles.

3.1. Meta-heurística ACO

La teoría de la meta-heurística ACO mencionada en esta sección fue presentada por Dorigo y Di Caro en [13], también aparece en el trabajo de Maniezzo et al. [24]. En los trabajos de Dorigo et al. [11] y Dorigo y Di Caro [12] fueron mencionadas ideas iniciales que luego constituyeron la teoría de la meta-heurística ACO.

Las hormigas artificiales se mueven aplicando una política estocástica de decisión local basada en dos parámetros, denominados rastros de feromonas τ y visibilidad (o atractividad) η . Los rastros de feromonas reflejan que los estados más visitados son altamente deseables, implementando de esta manera un proceso de auto catálisis o retroalimentación, mientras que la visibilidad refleja que estados menos costosos (evaluados según los objetivos a optimizar) son también deseables. La política de transición [13] se basa en asignar la probabilidad de ir del estado *i* al estado *j* según la ecuación:

$$p_{i,j}=\begin{cases}\dfrac{\tau_{i,j}{}^{\alpha}\eta_{i,j}{}^{\beta}}{\sum_{x\in J_i}\tau_{i,x}{}^{\alpha}\eta_{i,x}{}^{\beta}} & si\, j\in J_i\\ 0 & en\, caso\, contrario\end{cases} \tag{6}$$

donde J_i representa el vecindario de estados alcanzables desde el estado *i,* α *y* β son parámetros definidos a priori que reflejan la importancia relativa de las feromonas y la visibilidad respectivamente.

Al completar una solución o durante la construcción de la misma, la hormiga evalúa la solución y modifica los rastros de feromonas en las componentes de una matriz de feromonas τ que de esta forma guarda el conocimiento de las áreas ya exploradas. Esta información de feromonas guiará la búsqueda de futuras hormigas. El algoritmo también puede incluir un proceso de evaporación de rastros de feromonas [13], y otras acciones como realizar optimizaciones locales sobre soluciones encontradas o actualizar la información global para guiar el proceso de búsqueda desde una perspectiva no local. La actualización y evaporación de feromonas se realiza según la ecuación:

$$\tau_{i,j}=(1-\rho)\cdot\tau_{i,j}+\rho\cdot\Delta\tau \tag{7}$$

donde ρ representa el coeficiente de evaporación, y $\Delta\tau$ se calcula como:

$$\Delta\tau=\frac{1}{f_k(\vec{x})} \tag{8}$$

con $k\in\{1,2,\dots,b\}$ en el caso de realizar una actualización basada en un solo objetivo. Para el caso de actualización considerando los *b* objetivos al mismo tiempo [26], se calcula:

$$\Delta\tau=\frac{1}{\sum_{k=1}^{b}f_k(\vec{x})} \tag{9}$$

donde por motivos de normalización los valores f_k son divididos por un valor máximo definido a priori.

La hormiga que completó una solución debe actualizar el conjunto Pareto *CP* si la solución encontrada es no dominada con respecto a las existentes en *CP* y luego debe eliminar las soluciones dominadas por la misma. En la figura 1 se muestra el pseudo-código de un algoritmo ACO multi-objetivo genérico, denominado en adelante *MOACO (MultiObjective Ant Colony Optimization)*. Otros trabajos que presentan varios algoritmos ACO multi-objetivos pueden encontrarse en [16] y [17], mientras que en [23] se presentan varios diseños alternativos.

3.2 Multi-objective Ant Q Algorithm (MOAQ)

Este algoritmo se propuso en el trabajo de Mariano y Morales en [25]. La regla de transición de estados y la actualización de feromonas están basadas en el algoritmo Ant-Q, propuesto en [14]. Se mantiene una colonia de hormigas para cada objetivo. De esta manera en un problema con *b* objetivos tendremos *b* colonias de hormigas. Se mantiene una matriz de feromonas τ , y cada colonia actualiza esta matriz considerando su objetivo particular asignado.

La regla de transición de estados se basa en los valores de la matriz de feromonas τ , y la visibilidad η , la transición en cada colonia se realiza probabilísticamente según la ecuación (6). La actualización de los valores τ_{ij} se realiza basada en la ecuación (7). Se agrega como sumando al $\Delta\tau_{ij}$ la expresión $\gamma\cdot\max_{z\in J_j}\tau_{jz}$, donde γ representa el paso de apredizaje [14] y la expresión denota la calidad del camino según la información aprendida anteriormente.

3.3. BicriterionAnt (BiAnt)

Este algoritmo fue propuesto por Iredi et al. [21] para la resolución del TSP con dos objetivos y mantiene dos tablas de feromonas τ y τ' para los dos objetivos considerados. La distribución de probabilidades para seleccionar el siguiente estado está dada por:

$$p_{i,j}=\begin{cases}\dfrac{\tau_{i,j}^{\lambda\alpha}\cdot\tau'^{(1-\lambda)\alpha}_{i,j}\cdot\eta_{i,j}^{\lambda\beta}\cdot\eta'^{(1-\lambda)\beta}_{i,j}}{\sum_{x\in J_i}\tau_{i,x}^{\lambda\alpha}\cdot\tau'^{(1-\lambda)\alpha}_{i,x}\cdot\eta_{i,x}^{\lambda\beta}\cdot\eta'^{(1-\lambda)\beta}_{i,x}} & si\ j\in J_i\\ 0 & en\,caso\,contrario\end{cases} \tag{10}$$

Para cada hormiga t para $t\in\{1,2,...,m\}$, λ_t se calcula a partir de:

$$\lambda_t=\frac{t-1}{m-1} \tag{11}$$

De esta manera se consigue que las hormigas realicen búsquedas en distintas regiones del frente Pareto. La actualización y evaporación de feromonas se realiza para cada tabla según la ecuación (7).

```
procedure MOACO
inicializar_parametros()
while not condicion_parada()
      generacion=generacion + 1
      for ant=1 to m       // m es la cantidad de hormigas
             construir_solucion()
             evaluar_solucion()
             actualizar_feromonas() //según ecuación (7)
             actualizar_conjunto_pareto()
      end for
end while
end

procedure construir_solucion
sol={Ø}
while existen_estados_no_visitados()
      siguiente=seleccionar_siguiente_estado() // según (6)
      sol=sol ∪ {siguiente}
      marcar_como_visitado(siguiente)
      if(actualizacion_paso_a_paso)
             actualizar_feromonas_paso_a_paso()// según (7)
end while
end
```

Figura 1. Pseudo-código de un algoritmo ACO multi-objetivo genérico [13].

3.4. BicriterionMC (Bicriterion Multi Colony o BiMC)

Este algoritmo extiende el BicriterionAnt y también se presenta en [21], introduciendo la generalización de tener b colonias, cada una con su propia tabla de feromonas y visibilidad.

Utiliza la actualización por región para realizar la actualización de feromonas, en donde la secuencia de soluciones en el frente de soluciones no dominadas de la iteración se divide en b regiones de igual tamaño. Las hormigas que encontraron soluciones en la i-ésima región actualizan la colonia i, para $i\in[1,b]$. Este método fuerza de manera explícita a las colonias a buscar en diferentes regiones del frente Pareto.

3.5. Pareto Ant Colony Optimization (PACO)

Este algoritmo se propuso en [8] por Doerner et al., y se basa en la utilización de b tablas de feromonas (τ^b), una para cada objetivo. En cada iteración una hormiga computa una serie de pesos, que en [8] fueron seleccionados uniformemente aleatorios, $w=(w_1, w_{2,...}, w_b)$, y los utiliza al calcular la regla de transición de estados. Estando en el estado i se selecciona el estado j según:

$$j=\begin{cases} \max\limits_{j\in J_i}\left[\left(\sum_{k=1}^{b} w_k.\tau_{i,j}^{k}\right)^{\alpha}\cdot\eta_{i,j}^{\beta}\right] & si\, q<q_0 \\ \hat{i} & en\, caso\, contrario \end{cases} \tag{12}$$

donde $\hat{i}$ se calcula según la ecuación probabilística (6). Las dos mejores hormigas para cada objetivo actualizan la tabla de feromonas correspondiente a dicho objetivo según la ecuación (7).

Cada vez que una hormiga avanza a otro estado, se realiza una actualización local paso a paso de las b tablas de feromonas según la ecuación (7), considerando un valor constante para $\Delta\tau=\tau_o$, que representa el valor inicial para las feromonas (definido a priori).

3.6. MultiObjective Ant Colony System (MOACS)

MOACS, propuesto por Barán y Schaerer en [2], es una extensión del MACS-VRPTW, este último propuesto por Gambardella et al. [15]. Fue implementado considerando dos objetivos, utiliza una matriz de feromonas y dos visibilidades, una para cada objetivo del problema. La regla de transición de estados se calcula como:

$$j=\begin{cases} \max\limits_{j\in J_i}\left\{\tau_{i,j}\left[\eta_{i,j}^{0}\right]^{\lambda\beta}\left[\eta_{i,j}^{1}\right]^{(1-\lambda)\beta}\right\} & si\; q<q_0 \\ \hat{i} & en\, caso\, contrario \end{cases} \tag{13}$$

El cálculo de $\hat{i}$ se realiza según:

$$p_j=\begin{cases} \dfrac{\tau_{i,j}\left[\eta_{i,j}^{0}\right]^{\lambda\beta}\left[\eta_{i,j}^{1}\right]^{(1-\lambda)\beta}}{\sum\limits_{x\in J_i}\tau_{i,x}\left[\eta_{i,x}^{0}\right]^{\lambda\beta}\left[\eta_{i,x}^{1}\right]^{(1-\lambda)\beta}} & si\; j\in J_i \\ 0 & en\, caso\; contrario \end{cases} \tag{14}$$

Cada vez que una hormiga se mueve del estado i al estado j, realiza la actualización local de feromonas según la ecuación (7), con $\Delta\tau=\tau_o$, el valor inicial para las feromonas.

En el caso de encontrar una solución no dominada, se actualiza *CP* y se reinicializa la tabla de feromonas, considerando que la información fue aprendida por medio de soluciones dominadas [2]. Si la solución encontrada es dominada se realiza la actualización de feromonas según la ecuación (7).

3.7. Multiobjective Max-Min Ant System (M-MMAS o M3AS)

Este algoritmo, propuesto por Pinto et al. en [26], extiende el Max-Min Ant System [29] para resolver problemas multi-objetivos. Se utilizó inicialmente para resolver el problema de enrutamiento multicast multi-objetivo. Pinto et al. en su trabajo [26] optimizaron cuatro objetivos. Se mantiene una tabla de feromonas τ global que mantiene información de feromonas considerando todos los objetivos a optimizar. Una hormiga estando en el estado i escoge el siguiente estado a visitar, de acuerdo con la probabilidad p dada según la ecuación (14).

Las soluciones no dominadas actualizan la tabla de feromonas según la ecuación (7). Si $\tau_{ij}>\tau_{max}$, entonces $\tau_{ij}=\tau_{max}$, con $\tau_{max}=\Delta\tau^k/(1-\rho)$, y si $\tau_{ij}<\tau_{min}$, entonces $\tau_{ij}=\tau_{min}$, con $\tau_{min}=\Delta\tau^k/2m(1-\rho)$, donde m es el número de hormigas, k representa la k-ésima solución y $\Delta\tau^k$ se calcula con la ecuación (9). De esta manera se impone una cota inferior y otra superior al nivel de feromonas en los arcos.

3.8. COMPETants (COMP)

El algoritmo COMPETants, propuesto por Doerner et al. [9], fue utilizado en problemas bi objetivos, con dos tablas de feromonas y dos visibilidades. El número de hormigas para cada colonia no es fijo, y se asigna de manera dinámica durante la ejecución del algoritmo. Cada vez que todas las hormigas han construido sus soluciones, las colonias con las mejores soluciones reciben más hormigas para la siguiente iteración.

La probabilidad de transición de estados utiliza la ecuación (6). Se mantienen hormigas espías en cada colonia cuyas probabilidades de transición de estados utilizan información de ambas tablas de feromonas (ver el trabajo de Doerner et al. [9]). De esta manera se implementa un enfoque cooperativo entre las colonias. El algoritmo está basado en el *ASrank* [3], del cual mantuvo el enfoque de realizar un *ranking* y permitir a las mejores hormigas actualizar las tablas de feromonas, en este caso para cada objetivo.

3.9. Multiobjective Omicron ACO (MOA)

Este algoritmo fue propuesto por Gardel et al. en [18] y está basado en el algoritmo Omicron ACO propuesto por Barán y Gómez [19]. El algoritmo fue implementado considerando dos objetivos, utiliza una tabla de feromonas y dos visibilidades, una para cada objetivo del problema.

El algoritmo construye soluciones, las evalúa respecto a los objetivos del problema y guarda una población P de soluciones no dominadas hasta ese momento. Luego de K iteraciones, donde K es un parámetro definido a priori, se utilizan todas las soluciones del conjunto P para actualizar la tabla de feromonas. Antes de realizar la actualización se reinicializa la tabla de feromonas a τ_0 con $\tau_0 > 0$. El parámetro O (*ómicron*) determina la cantidad de feromonas que cada individuo deposita. La regla de actualización de feromonas es como sigue:

$$\tau_{i,j} = \tau_{i,j} + O/h \tag{15}$$

donde h es la cardinalidad del conjunto P. El nivel de feromonas queda acotado así como con el *MMAS* [29]. Al construir las soluciones, la probabilidad de ir del estado i al estado j está dada por la ecuación (14).

4. Multiobjective Ant System (MAS)

El *MAS* que propone este trabajo es una extensión sencilla del *AS* (*Ant System*) [11] para manejar múltiples objetivos. Las modificaciones realizadas se centran en la selección del siguiente estado y la actualización de las feromonas. De esta manera se mantiene una única tabla de feromonas con una visibilidad para cada objetivo a optimizar, y las hormigas se distribuyen en regiones del espacio de búsqueda por medio del valor λ utilizado, según la regla (14) ya mencionada en el *MOACS*.

La actualización de feromonas se realiza luego de construir todas las soluciones de la iteración y es realizada por las soluciones no dominadas encontradas en la iteración. La regla de actualización utilizada es la ecuación (7) con $\Delta\tau$ según la ecuación (9), de esta manera la actualización queda en función a la evaluación de la solución con todos los objetivos del problema.

Se agregó un mecanismo de control de convergencia al algoritmo, que consiste en reiniciar la tabla de feromonas si durante K' generaciones, con K' definido a priori, no se encontraron soluciones no dominadas. De esta manera, se favorece la exploración de nuevos caminos. Este mecanismo ayuda a continuar la búsqueda ante situaciones de convergencia como óptimos locales o el problema denominado *stagnation behavior*, mencionado en la implementación del *AS* [11], en el cual se terminaba el algoritmo ante esta situación. En la nueva propuesta, se reinicializa la tabla de feromonas de manera a continuar la ejecución. El pseudo-código del MAS se muestra en la figura 2.

```
procedure MAS
inicializar_parametros()
while not condicion_parada()
      generacion=generacion + 1
      repeat para cada hormiga k
             construir_solucion()        // según ecuación (14)
             actualizar_frente_pareto()
             actualizar_feromonas()      // según ecuación (7)
      end repeat
      if (no_cambio(CP,K')) //sin cambios en K' generaciones
             reiniciar_feromonas()
end while
end
```

Figura 2. Pseudo-código del MAS

5. Resultados experimentales

Todos los algoritmos fueron implementados en C++ (*Compilador GNU GCC*) y fueron ejecutados en un entorno Linux, distribución Mandrake 9, en una máquina AMD 1733 MHz con 512 MB de memoria. Se realizaron diez corridas de 200 segundos para cada algoritmo y para cada problema de prueba. Como problemas de prueba se utilizaron dos instancias de cada tipo de problema (TSP, QAP y VRPTW). En el caso del TSP se utilizaron las instancias bi-objetivas de 100 ciudades KROAB100 y KROAC100, que se encuentran en la distribución del TSPLIB (http://www-idss.cs.put.poznan.pl/~jaszkiewicz). Para el QAP bi-objetivo, se utilizaron las instancias de 75 localidades qapUni.75.0.1 y qapUni.75.p75.1 que pueden ser encontradas en [27] o en el sitio web http://www.intellektik.informatik.tudarmstadt.de/~lpaquete/QAP. Para el VRPTW bi-objetivo se utilizaron las instancias de 100 clientes c101 y rc101 tomados del conjunto de 56 problemas de prueba propuestos por Solomon en [28]. Se utilizó m=10 hormigas, $\alpha=1$, $\beta=2$, $\gamma=0.3$ (*MOAQ*), $\rho=0.1$, $\lambda=0.8$ (*MOAQ*), $\tau_{max}=0.9$ (*MOAQ*), $O=10$ (*MOA*), $K=100$ *y* $K'=500$ (*MOA y MAS* respectivamente), $q_0=0.5$, $\tau_0=1$.

5.1. Métricas de desempeño

Se utilizaron 4 métricas de evaluación del desempeño de los algoritmos, de manera a realizar un análisis comparativo entre los mismos. Las métricas denominadas *M1'*, *M2'*, y *M3'*, tomadas de Zitzler et al. [30], se refieren respectivamente a la evaluación de la calidad, distribución y extensión del frente Pareto generado por el algoritmo. La cuarta métrica denominada Error fue tomada de Veldhuizen [31] y se refiere al porcentaje de soluciones generadas que no pertenecen al frente Pareto.

En todos los casos se utiliza como métrica de distancia, la distancia euclidiana entre dos puntos, representada por $d(p,q)$, y se considera Y_{true} el frente Pareto conocido e Y' el frente Pareto generado por los algoritmos. Las cuatro métricas utilizadas se definen a continuación.

$$M1'(Y')=\frac{1}{|Y'|}\sum_{p\in Y'} min\{d(p,\bar{p}) \mid \bar{p}\in Y_{true}\}$$

donde $|\cdot|$ representa cardinalidad.

La métrica *M2'* utiliza un conjunto $W=\{q\in Y' \mid d(p,q)>\sigma\}$, dado un valor de distancia σ dependiente del problema, y se calcula como:

$$M2'(Y')=\frac{1}{|Y'|-1}\sum_{p\in Y'}|W|$$

La métrica *M3'* calcula la extensión del frente Pareto generado según:

$$M3'(Y')=\sqrt{\sum_{i=1}^{b} max\{d(p_i,q_i) \mid p,q\in Y'\}}$$

La métrica de Error se calcula como:

$$E=\frac{\sum_{i=1}^{|Y'|} e_i}{|Y'|}$$

donde e_i toma el valor 0 si la *i*-ésima solución de Y' pertenece al Y_{true} , 1 en caso contrario.

Las métricas *M1'*, *M2'* y *M3'* fueron normalizadas con respecto a un valor máximo de las mismas, dependiendo del problema particular. De esta manera, los resultados obtenidos representan porcentajes que se utilizan para comparar los diversos algoritmos en el conjunto de problemas de prueba.

Para la métrica *M2'* se utilizó como valor para σ el diez por ciento de la distancia entre el punto con mejor evaluación en el primer objetivo y el punto con mejor evaluación en el segundo objetivo del frente Y_{true} , así σ representa el diez por ciento de la distancia entre los puntos extremos del Y_{true} para cada problema particular.

5.2. Resultados de las ejecuciones

El frente Y_{true} conocido de cada problema fue generado previamente tomando las soluciones no dominadas generadas por todos los algoritmos en varias corridas [26]. Las tablas a continuación muestran los resultados promedios de las evaluaciones de las métricas aplicadas a los frentes Pareto generados por los algoritmos antes descriptos para cada problema.

En las tablas 1 al 3 se muestran de manera comparativa los valores promedio en 10 corridas de las métricas aplicadas a los frentes generados por cada algoritmo. Para los problemas del tipo VRPTW no se tomaron en cuenta las métricas *M2'* y *M3'*, que evalúan la distribución y extensión del frente obtenido, debido a que los frentes Pareto encontrados contaban con pocas soluciones no dominadas.

Tabla 1. *Ranking* ordenado del promedio de métricas para el TSP.

M1’	Alg.	M2’	Alg.	M3'	Alg.	Error	Alg.
0,0217	PACO	0,8212	MOACS	1,0000	COMP	0,8575	PACO
0,0442	MAS	0,8167	M3AS	0,9871	MOAQ	0,9309	MAS
0,0610	M3AS	0,8138	MOA	0,9562	MOACS	0,9858	M3AS
0,0625	MOACS	0,7800	MAS	0,9371	MOA	0,9924	MOACS
0,0709	MOA	0,7710	MOAQ	0,9139	M3AS	0,9925	BiMC
0,1333	BiMC	0,7233	BiANT	0,8842	MAS	0,9980	BiANT
0,1500	BiANT	0,6439	BiMC	0,7239	BiANT	0,9995	MOA
0,1941	COMP	0,6374	COMP	0,6185	BiMC	1,0000	COMP
0,4225	MOAQ	0,0641	PACO	0,1318	PACO	1,0000	MOAQ

Tabla 2. *Ranking* ordenado del promedio de métricas para el QAP.

M1’	Alg.	M2’	Alg.	M3'	Alg.	Error	Alg.
0,2631	MOACS	0,6950	MOAQ	0,9069	MOAQ	0,8677	MOACS
0,2791	M3AS	0,6770	COMP	0,8984	MOACS	0,9798	M3AS
0,3796	MAS	0,6581	PACO	0,8966	COMP	0,9875	COMP
0,3837	COMP	0,6341	BiMC	0,8884	PACO	0,9937	MOA
0,3876	MOA	0,6297	M3AS	0,8588	M3AS	0,9950	BiANT
0,4092	PACO	0,5841	BiANT	0,7531	BiANT	0,9962	MAS
0,4521	MOAQ	0,5745	MOACS	0,7399	BiMC	0,9972	PACO
0,4607	BiMC	0,4852	MOA	0,7258	MOA	1,0000	BiMC
0,5439	BiANT	0,4383	MAS	0,6842	MAS	1,0000	MOAQ

En ambos problemas TSP, el MAS obtuvo el frente más cercano al óptimo en grandes sectores, solo superado en pequeños sectores centrales por el PACO, que obtuvo buenas soluciones pero sin distribución ni extensión (*M2'* y *M3'*). El MOACS y el M3AS obtuvieron buenos resultados promedios para el TSP. En los problemas QAP (tabla 2) se puede notar nuevamente el buen comportamiento promedio del MOACS y el M3AS.

Tabla 3. *Ranking* ordenado del promedio de métricas para el VRPTW

M1'	Alg.	Error	Alg.
0,057136	MAS	0,850000	BIANT
0,067716	BIANT	0,9500050	MAS
0,080535	MOACS	1,000000	BIMC
0,094266	COMPET	1,000000	COMPET
0,149011	BIMC	1,000000	M3AS
0,210819	PACO	1,000000	MOA
0,285776	M3AS	1,000000	MOACS
0,419749	MOA	1,000000	MOAQ
0,731304	MOAQ	1,000000	PACO

Tabla 4. Ranking ordenado de promedios generales de 60 corridas de cada algoritmo

M1’	Alg.	M2’	Alg.	M3'	Alg.	Error	Alg.
0,1354	MOACS	0,5864	MOAQ	0,7586	COMP	0,9476	BIANT
0,1603	MAS	0,5785	M3AS	0,7576	MOAQ	0,9515	PACO
0,2086	M3AS	0,5583	MOACS	0,7418	MOACS	0,9534	MOACS
0,2139	PACO	0,5258	COMP	0,7091	M3AS	0,9590	MAS
0,2240	COMP	0,5230	BIANT	0,6652	MOA	0,9885	M3AS
0,2477	BIMC	0,5196	MOA	0,6273	MAS	0,9958	COMP
0,2539	BIANT	0,5112	BIMC	0,5908	BIANT	0,9975	BIMC
0,2927	MOA	0,4873	MAS	0,5434	BIMC	0,9977	MOA
0,5353	MOAQ	0,2889	PACO	0,4081	PACO	1,0000	MOAQ

En las instancias del problema VRPTW, el MAS y el BicriterionAnt obtienen los mejores resultados, además el MOACS obtiene buenas soluciones. La tabla 4 presenta un promedio de los valores de las métricas para los 6 problemas por cada algoritmo, ordenados desde el mejor algoritmo hasta el peor, para cada métrica. Como se muestra en la tabla 4, el MOACS presenta el mejor promedio de distancia relativa al frente Pareto, métrica *M1’*, lo cual empíricamente demuestra que el MOACS produjo en promedio frentes Pareto más cercanos al óptimo que los demás algoritmos. En las demás métricas del promedio general, el MOACS obtuvo resultados considerablemente cercanos al primero en cada métrica, lo cual teniendo en cuenta su promedio en la métrica *M1’,* sugiere al MOACS como el mejor algoritmo en promedio.

El algoritmo propuesto en este trabajo, el MAS, obtuvo el segundo lugar en distancia promedio relativa al frente Pareto, lo cual es significativo atendiendo que supera a 7 algoritmos ACO multi-objetivos del estado del arte en cuanto a calidad de las soluciones obtenidas. En los problemas específicos del tipo VRPTW obtuvo la menor distancia promedio con respecto a los frentes Pareto superando incluso al MOACS (tabla 3).

6. Conclusiones y trabajos futuros

Se presentó una comparación exhaustiva entre 8 algoritmos que componen el estado del arte en la optimización multi-objetivo basada en colonia de hormigas y 1 algoritmo propuesto en este trabajo, considerando un representativo conjunto de problemas de prueba, clásicos en la literatura de optimización combinatoria. Empíricamente se verificó que el MOACS presenta el mejor comportamiento general atendiendo el conjunto propuesto de problemas de prueba.

Atendiendo a los resultados generales en promedio, se puede notar que al considerar la distancia al frente óptimo los tres mejores algoritmos implementan la estrategia de utilizar una única tabla de feromonas y varias visibilidades dependientes de cada objetivo a optimizar. Esto empíricamente demuestra que la estrategia de usar una única matriz de feromonas con varias visibilidades obtiene un buen comportamiento con respecto a otras posibles implementaciones en la optimización multi-objetivo.

Como trabajo futuro se propone buscar mejorar la implementación del MAS con el objetivo de volverlo más competitivo, ya que la implementación actual es una versión muy sencilla del *Ant System* al cual se pueden agregar estrategias específicas de optimización como optimizadores locales o mejorar la técnica de control de convergencia implementando un mecanismo más refinado que la simple reinicialización de feromonas que representan los conocimientos aprendidos. Considerando los resultados obtenidos en este trabajo, el MAS propuesto obtuvo un buen comportamiento general, por lo cual puede ser interesante seguir investigando su comportamiento.

También se puede extender este trabajo con el fin de comparar los algoritmos MOACO con más problemas de prueba multi-objetivos. Inclusive se puede realizar una implementación de un algoritmo en equipo distribuido [1] utilizando los mejores algoritmos MOACO y comparar su comportamiento con los resultados obtenidos en este trabajo.

Referencias

[1] B. Barán, E. Kaszkurewicz y A. Bhaya. "Parallel Asynchronous Team Algorithms: Convergence and Performance Análisis". IEEE Transactions on Parallel & Distributed Systems, Vol. 7, No. 7, pg. 677-688, julio 1996. Estados Unidos.

[2] B. Baran y M. Schaerer. "A multiobjective Ant Colony System for Vehicle Routing Problems with Time Window*s".* Proc. Twenty first IASTED International Conference on Applied Informatics, pg. 97-102. Insbruck, Austria. 2003.

[3] B. Bullnheimer, R. Hartl y C. Strauss. "A new rank based version of the Ante System. A computational study". Central European Journal for operations Research and Economics, 7:1. 25-38. 1999.

[4] E. Çela. "The Quadratic Assignment Problem. Theory and algorithms". Kluwer Academic Publishers, 1998.

[5] C. Coello. "An updated Survey of Evolutionary Multiobjective Optimization Techniques, state of the art and future trends". In Congress on Evolutionary Computation. Piscataway, N. J. IEEE Service Center. 3–13. 1999.

[6] K. Deb. "Evolutionary Algorithms for Multi-Criterion Optimization in Engineering Design". In Proceedings of Evolutionary Algorithms in Engineering and Computer Science EUROGEN'99. 1999.

[7] J. Denenbourg, S. Aron, S. Goss y Pasteels. "The self-organizing exploratory pattern of the argentine ant". Journal of Insect Behavior, 3:159–168. 1990.

[8] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss y C. Stummer. "Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection". Proceedings of the 4th. Metaheuristics International Conference. Porto, 243-248. 2002.

[9] K. Doerner, R. Hartl y M. Reimann. "Are COMPETants more competent for problem solving? – the case of a multiple objective transportation problem". Central European Journal of Operations Research, 11:2, 115-141. 2003.

[10] M. Dorigo y L. Gambardella. "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem". IEEE Transactions on Evolutionary Computation, 1:1, 53-66. 1997.

[11] M. Dorigo, V. Maniezzo y A. Colorni. "The Ant System: Optimization by a colony of cooperating agents". IEEE Transactions on Systems, Man, and Cybernetics - Part B, 26:1, 29-41. 1996.

[12] M. Dorigo y G. Di Caro. "Ant Algorithms for discrete optimization". Artificial Life, 5(2):137–172. 1999.

[13] M. Dorigo y G. Di Caro. "The Ant Colony Optimization Meta-Heuristic". In D. Corne, M. Dorigo, and F. Glover (Eds.), New Ideas in Optimization, McGraw Hill, London, UK, 11-32. 1999.

[14] L. Gambardella y M. Dorigo. "Ant-Q: A reinforcement Learning approach to the traveling salesman problem". In A. Prieditis and S. Russell, editors, Proceedings of the Twelfth International Conference on Machine Learning (ML-95), 252–260. Morgan Kaufmann Publishers, Palo Alto, CA. 1995.

[15] L. Gambardella, E. Taillard y G. Agazzi. "MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows". In D. Corne, M. Dorigo, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, 73-76. 1999.

[16] C. García-Martínez, O. Cordón y F. Herrera. "An Empirical Análisis of Multiple Objective Ant Colony Optimization Algorithms for the Bi-criteria TSP". ANTS Workshop 61-72. 2004.

[17] C. García-Martínez, O. Cordón y F. Herrera. "A Taxonomy and an empirical anlysis of Multiple Objective Ant Colony Optimization Algorithms for the Bi-criteria TSP". Technical Report SCI2S-2004-12. Research Group on Soft Computing and Intelligent Information Systems. University of Granada. 2004.

[18] P. Gardel, H. Estigarribia, U. Fernández y B. Barán. "Aplicación del Ómicron ACO al problema de compensación de potencia reactiva en un contexto multiobjetivo". Congreso Argentino de Ciencias de la Computación - CACIC'2005. Concordia – Argentina. 2005.

[19] O. Gómez y B. Barán. "Omicron ACO". Proceedings of CLEI'2004. Latin-American Conference on Informatics (CLEI). Arequipa. Perú. 2004.

[20] S. Goss, S. Aron, S. Denenbourg y J. Pasteels. “Self-organized shortcuts in the Argentine ant”. Naturwissenschaften, 76:579–581. 1989.
[21] S. Iredi, D. Merkle y M. Middendorf. “Bi-Criterion Optimization with Multi Colony Ant Algorithms”. Proc. First International Conference on Evolutionary Multi-criterion Optimization (EMO'01), Lecture Notes in Computer Science 1993, 359-372. 2001.
[22] J. Knowles y D. Corne. “Instance generators and test suites for the multiobjective quadratic assignment problem”. In: Fonseca, C.M., et al. Editors. Proc of EMO '03, LNCS 2632 page 295-310, Springer-Verlag, 2003.
[23] M. Lopez-Ibañez, L. Paquete y Stutzle. “On the design of ACO for the Biobjective Quadratic Assignment Problem”. En: Dorigo, M., Birattari, M., Blum, C., Gambardella, L., Montada, F., Stützle, T. (Eds.): Proc. of the Fourth International Workshop on Ant Colony Optimization (ANTS 2004), Lecture Notes in Computer Science, Springer Verlag. 2004.
[24] V. Maniezzo, L. Gambardella y F. de Luigi. “Ant Colony Optimization”. En: Onwubolu, G. C., and B. V. Babu (Eds.): New Optimization Techniques in Engineering, Springer-Verlag, Berlin Heidelberg, pg. 101-117. 2004.
[25] C. Mariano y E. Morales. “A Multiple Objective Ant-Q Algorithm for the Design of Water Distribution Irrigation Networks”. Technical Report HC-9904, Instituto Mexicano de Tecnología del Agua, Mexico, June, 1999.
[26] D. Pinto y B. Barán. “Solving Multiobjective Multicast Routing Problem with a new Ant Colony Optimization approach”. LANC’05, Cali, Colombia. 2005.
[27] S. Sahni y T. González. “P-complete aproximation problems”. Journal of the ACM, 23:555-565, 1976.
[28] M. Solomon. “Algorithms for the vehicle routing and scheduling problem with time window constraints”. Operations Research 35. 254-365, 1987.
[29] T. Stutzle y H. Hoos. “Max-Min Ant System”. Future Generation Computer Systems, 16:8, 889-914. 2000.
[30] E. Zitzler, K. Deb y L. Thiele. “Comparison of multiobjective evolutionary algorithms. Empirical result”, evolutionary computation. 8:2, pp 173-195, 2000.
[31] D. A. van Veldhuizen. “Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations”. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, Mayo 1999.