Fuzzy Multiobjective Particle Swarm Optimization applied to the Next Release Problem

Optimización Multiobjetivo Difusa mediante Enjambre de Partículas aplicada al Problema del Próximo Lanzamiento

Carlos Casanova, Giovanni Daián Rottoli, Esteban Schab, Anabella De Battista, Adrián Tournoud, Luciano Bracco, Fernando Pereyra Rausch

Grupo de Investigación sobre Inteligencia Computacional e Ingeniería de Software
Departamento de Ingeniería en Sistemas de Información
Facultad Regional Concepción Del Uruguay
Universidad Tecnológica Nacional (UTN)
Ing. Pereira 676, Concepción del Uruguay (CP 3260), Entre Ríos, Argentina

{casanovac,rottolig,schabe,debattistaa,tournouda,braccol,pereyraf\}
@frcu.utn.edu.ar

Abstract. *In this paper a novel metaheuristic method for multi-objective optimization based on Particle Swarms and Fuzzy Logic is proposed: the Fuzzy Multi-Objective PSO (FMOPSO). An implementation for this method is presented to approach a classic Search Based Software Engineering Problem: the Next Release Problem (NRP). A proof of concept is carried out applying this algorithm to a bi-objective instance of the aforemetioned problem and comparing it with another state-of-the-art metaheuristic. Finally, the conclusion highlights the most relevant results.*
Keywords: *Multiobjective Optimization, Particle Swarm Optimization, Next Release Problem, Fuzzy Logic.*

Resumen. *En este trabajo se presenta un método novedoso basado en Enjambres de Partículas y Lógica Difusa para optimización multiobjetivo: el FMOPSO (Fuzzy Multi-Objective Particle Swarm Optimization). Este método se presenta en el contexto de la resolución de un problema clásico de la Ingeniería de Software Basada en Búsqueda: el Problema del Próximo Lanzamiento (Next Release Problem). Se realiza una prueba de concepto aplicando este algoritmo a una instancia bi-objetivo del problema mencionado anteriormente, y se lo compara con otra metaheurística del estado del arte. Finalmente, se concluye resaltando los resultados más importantes.*
Palabras clave: *Optimización Multi-Objetivo, Optimización por Enjambre de Partículas, Problema del Próximo Lanzamiento, Lógica Difusa.*

1. Introducción

En primer lugar, en la presente sección introductoria, se describe la Ingeniería de Software Basada en Búsqueda (*Search Based Software Engineering, SBSE*), enumerando sus características particulares en la sección 1.1, para luego abordar el Problema del Próximo Lanzamiento (*Next Release Problem, NRP*) en la sección 1.2.

1.1. Ingeniería de Software Basada en Búsqueda

La Ingeniería de Software estudia los problemas relacionados al desarrollo, la operación y el mantenimiento del software desde un punto de vista científico. Aun así, en muchos ámbitos de la academia y la industria, el software se sigue construyendo de una forma muy cercana a la artesanal, dependiendo en gran parte de la experticia y el buen juicio del ingeniero el lograr entregables y piezas de software que, ante todo, cumplan con los requerimientos, pero además que sea propenso al bajo impacto al cambio, alta cohesión, bajo acoplamiento, entre otras características deseables. En muchas ocasiones no es sencillo determinar un camino para lograr estas características, es decir, se vuelve difícil determinar cómo construir, operar o mantener el software para que éste sea un producto de calidad.

La Ingeniería de Software Basada en Búsqueda (*Search Based Software Engineering,* SBSE) es una disciplina que busca brindar soporte al ingeniero de software para lograr productos de calidad mediante la aplicación de métodos de búsqueda. Este nuevo conjunto de prácticas mueve el foco de la descripción de los pasos para la obtención del producto a la descripción de las características deseadas del mismo.

Esta descripción de las características de los productos de la ingeniería de software debe ser codificada para ser entendida por un método de búsqueda, usualmente del tipo metaheurístico, que es capaz de generar productos posibles y evaluar su calidad. Mediante un conjunto de reglas, el método genera soluciones (es decir, posibles productos) que son comparadas y refinadas en cada iteración. En consecuencia, es el método de búsqueda el que se encarga de determinar el cómo obtener el producto.

Los problemas resueltos por este tipo de enfoques son formulados como problemas de optimización que suelen ser combinatorios. Por este motivo los métodos de búsqueda suelen ser de naturaleza metaheurística, descartándose en la mayoría de los casos los métodos de solución basados en optimización clásica como la programación matemática.

1.2. Problema NRP

Un aspecto clave de cualquier proyecto de desarrollo de software de grandes dimensiones es determinar un conjunto de requerimientos que satisfaga a las partes interesadas (*stakeholders*). El problema del próximo lanzamiento (*next release problem*, NRP), originalmente propuesto por Bagnall y otros [Bagnall et al. 2001], formaliza esta cuestión. Consiste en hallar un subconjunto de requerimientos, o un subconjunto de *stakeholders*, que maximice un atributo deseado, tal como el beneficio, sujeto a una cota superior de algún otro atributo que usualmente es el costo. Eventualmente, este problema también puede estar restringido por dependencias entre los requerimientos: precedencia, simultaneidad, elegir no más que k de n requerimientos, entre otros.

Este problema NRP, denominado de ahora en más **mono-objetivo**, es reducible al problema de la mochila (*knapsack problem*) [Bagnall et al. 2001] y constituye, por lo tanto, un problema de optimización combinatoria NP-*hard*. Esto implica que no existe algoritmo de tiempo polinomial que lo resuelva, a menos que $P = NP$. Esta clase de problemas es a menudo atacada por técnicas de búsqueda heurística o metaheurística debido al excesivo esfuerzo requerido para hallar soluciones exactas.

Ejemplos del uso de estas herramientas se puede evidenciar en trabajos como [Jiang et al. 2010], en el cual se propone el uso de un algoritmo híbrido basado en colonias de hormigas (HACO) para abordar el problema, o en la propuesta de [Xuan et al. 2012], que utiliza un algoritmo *Backbone-based* multinivel para resolver el problema de la escalabilidad. Por otro lado, algoritmos genéticos como NSGA-II y MOCell, así como el evolutivo PAES, han sido probados en escenarios multiobjetivo [Zhang et al. 2007, Durillo et al. 2009, Durillo et al. 2011] (cuestión que será desarrollada en futuras secciones).

1.2.1. Formulación lineal del NRP Mono-objetivo

El problema NRP mono-objetivo puede formularse como un modelo de Programación Lineal Entera. En esta formulación se asume que el tomador de decisiones desea maximizar el beneficio sujeto a una restricción de costo.

Sean $X = [x_1, x_2, \dots, x_n]$ e $Y = [y_1, y_2, \dots, y_m]$ los vectores binarios de decisión que representan la inclusión de los requerimientos, del 1 al *n*, y la satisfacción de los *stakeholders*, del 1 al *m*.

Sea $C = [c_1, c_2, \dots, c_n]$ el vector de costos asociados a los requerimientos, $B = [b_1, b_2, \dots, b_m]$ el vector de los beneficios asociados a cada *stakeholder*, y *p* una cota para el costo.

Sea *P* la relación de precedencia formada por los pares (*i*, *j*), tales que el requerimiento *i* debe ser seleccionado siempre que se selecciona el requerimiento *j*.

Sea además *I* la relación de interés formada por los pares (*i*, *k*), donde el *stakeholder k* ha solicitado el requerimiento *i*.

Así, se puede modelar el problema NRP de la siguiente forma:

$$maxf(Y) = \sum_{i=1}^{m} b_i \cdot y_i \tag{1a}$$

sujeto a

$$\sum_{j=1}^{n} c_j \cdot x_j \leq p \tag{1b}$$

$$x_i \geq x_j, \forall (i,j) \in P \tag{1c}$$

$$x_i \geq y_k, \forall (i,k) \in I \tag{1d}$$

$$X \in \{0,1\}^n, Y \in \{0,1\}^m \tag{1e}$$

Esta es una formulación general, donde se requiere la satisfacción de **todos** los requerimientos asociados al *stakeholder* para computar su beneficio asociado [Veerapen et al. 2015].

1.2.2. Formulación lineal del NRP Bi-objetivo

En la versión bi-objetivo de NRP [Zhang et al. 2007], en contraste, la restricción de cota superior del costo se transforma en un segundo objetivo, el cual usualmente se encuentra en conflicto con el primero. Como resultado, se le presenta al tomador de decisiones un conjunto de soluciones potenciales denominado frente Pareto-óptimo, que representa las posibilidades de compromiso entre los dos objetivos. Este enfoque permite al tomador de decisiones seleccionar una solución de acuerdo a sus preferencias.

Adicionalmente, el frente de Pareto puede proporcionar información valiosa sobre el resultado del conjunto de requisitos seleccionado para el tomador de decisiones, ya que captura las concesiones entre los dos objetivos en competencia. Los resultados también se pueden utilizar para evaluar cuestiones de tipo "¿qué pasa si…?".

Sea X, Y, C, B, P e I, tal como se han definido en la formulación mono-objetivo del problema NRP presentada previamente, el problema bi-objetivo se define como sigue:

$$maxf(Y) = \sum_{i=1}^{m} b_i \cdot y_i \tag{2a}$$

$$ming(X) = \sum_{j=1}^{m} c_j \cdot x_j \tag{2b}$$

sujeto a

$$x_i \geq x_j, \forall (i,j) \in P \tag{2c}$$

$$x_i \geq y_k, \forall (i,k) \in I \tag{2d}$$

$$X \in \{0,1\}^n, Y \in \{0,1\}^m \tag{2e}$$

El resto del trabajo se estructura de la siguiente forma: en la sección 2 se describen brevemente los algoritmos metaheurísticos basados en enjambres de partículas; la sección 3 introduce consideraciones importantes respecto de la optimización cuando varios objetivos se ven involucrados; se propone en la sección 4 el algoritmo *Fuzzy Multi-Objective PSO* (FMOPSO), utilizado para aproximar el frente Pareto-óptimo de problemas bi-objetivo, utilizándose particularmente en el NRP; en la sección 5 se presenta una prueba de concepto del algoritmo que constituye el primer estudio realizado para tomar conocimiento sobre la habilidad de este algoritmo de búsqueda, comparándolo con otro algoritmo del estado del arte que cumple la misma función: el *Multi-Objective PSO* (MOPSO). Por último, en la sección 6 se presentan las conclusiones de este trabajo.

2. Optimización por Enjambre de Partículas

En el presente apartado se introduce el algoritmo clásico de optimización basado en enjambres de partículas (*Particle Swarm Optimization*, PSO) en la sección 2.1, se

presentan distintas estrategias o topologías de comunicación entre las partículas del enjambre en la sección 2.2, y finalmente se detalla el PSO Canónico en la sección 2.3.

2.1. PSO Clásico

La Optimización por Enjambre de Partículas (*Particle Swarm Optimization*, PSO) es una metaheurística poblacional, esto es, mantiene en cada iteración un conjunto (enjambre) de soluciones candidatas llamadas partículas, las cuales se mueven por el espacio de soluciones.

El movimiento, es decir, el cambio de posición de cada partícula, se realiza mediante una ecuación de movimiento. La regla de movimiento especifica la forma de perturbación de la solución, por lo que esta metaheurística es también del tipo perturbativa (en contraposición a las constructivas).

Cada partícula cuenta con acceso a una **función de aptitud**, la cual constituye una **medida de eficiencia** de la posición actual de cada partícula, y es el objetivo que se desea optimizar. Brevemente, cada partícula i en la iteración $k+1$ cambia su posición $X_i^{[k+1]}$ aplicando la velocidad $V_i^{[k+1]}$, calculada según la regla de movimiento:

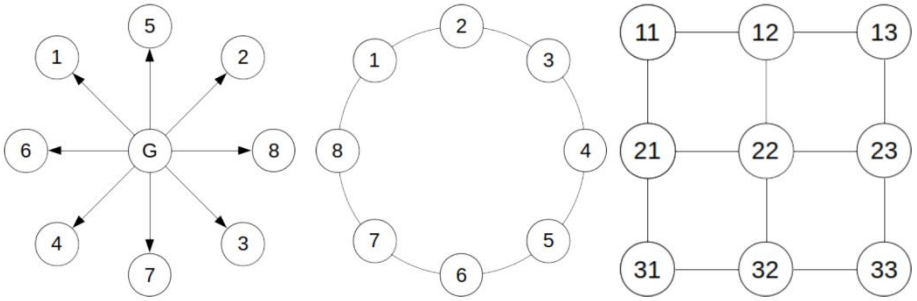
$$V_i^{[k+1]} = V_i^{[k]} + w_C \times r_1^{[k+1]} \times \left[b_i^{[k]} - X_i^{[k]}\right] + w_S \times r_2^{[k+1]} \times \left[b_G^{[k]} - X_i^{[k]}\right] \quad (3)$$

donde $b_i^{[k]}$ simboliza la mejor posición alcanzada por la partícula en iteraciones anteriores, $b_G^{[k]}$ es la mejor posición alcanzada por todo el enjambre en iteraciones anteriores, r_1 y r_2 son números aleatorios distribuidos uniformemente en el intervalo [0,1], y w_C y w_S son constantes predeterminadas (parámetros del algoritmo). Los tres términos de la ecuación de movimiento representan, en orden, inercia, memoria y cooperación. La posición $X_i^{[k+1]}$ se modifica de la siguiente manera:

$$X_i^{[k+1]} = X_i^{[k]} + V_i^{[k+1]} \quad (4)$$

2.2. Topologías de comunicación

Las primeras versiones de PSO abrieron el estudio de los grafos de influencia o topologías de comunicación entre las partículas. Una topología de comunicación o vecindario para las partículas se define como una función $s: S \rightarrow 2^S$, siendo S el conjunto de todas las partículas, y 2^S el potencial o conjunto de partes de dicho conjunto. La función $s(p)$ asigna a cada partícula un subconjunto de la población total de partículas. Tal subconjunto es llamado **vecindario** de la partícula.



(a) Estrella (b) Anillo (c) Grilla

Figura 1. Ejemplos de topologías de comunicación entre partículas

La utilidad de la topología estriba en regular la información disponible para cada partícula. Esta información es la que se utiliza en el tercer término de la ecuación de movimiento: cooperación. Además, debe existir un criterio según el cual cada partícula, de acuerdo al subconjunto de vecinos que la topología reinante determine, construye el término cooperación de la ecuación de movimiento. Este criterio generalmente se basa en los valores de la función de aptitud de las posiciones b_j, para todas las partículas j en el vecindario de la partícula en cuestión.

De ellas, la topología llamada Estrella Global es la utilizada por el PSO Clásico: la partícula que posee la mejor posición del enjambre hasta la iteración k, $b_G^{[k]}$, la informa al resto. Otros ejemplos de topologías pueden encontrarse en las referencias [Kennedy y Mendes 2003, Clerc 2013] entre las cuales se pueden destacar las variantes Estrella Estocástica, Estrella Estocástica Global/Individual, Radial, *Grid*, y Von Neumann.

2.3. PSO Canónico

PSO es un algoritmo originalmente formulado para trabajar con variables continuas. Sin embargo, es posible utilizarlo para problemas de variable discreta mediante el uso de métodos de confinamiento [Clerc 2006] que eviten el escape del espacio de soluciones factibles del problema. Este tipo de métodos puede involucrar el redondeo a la variable entera más próxima, o alguna regla de "rebote" cuando la partícula llega a un extremo del espacio de soluciones.

Aunque la utilización de métodos de confinamiento puede resultar en muchos casos adecuada (basta consultar referencias como [Schweickardt et al. 2016, Camargo et al. 2018]), en algunos problemas de optimización combinatoria la función de aptitud resulta demasiado complicada como para que las partículas puedan encontrar su camino hacia una buena solución. En estas ocasiones puede reformularse el algoritmo PSO para tomar en cuenta los aspectos combinatorios dentro de la ecuación de movimiento. Es así que surge el modelo PSO Canónico [Clerc 2006], el cual determina condiciones necesarias y suficientes para poder utilizarse en cualquier dominio. Según este modelo, se debe definir:

(a) la representación de las soluciones, es decir, las posiciones y las velocidades posibles;
(b) una función de aptitud, que puede ser mono-valuada o multi-valuada;
(c) una relación de orden (parcial o total) sobre el recorrido de la función de aptitud;

(d) las operaciones binarias
- *resta*(posición, posición)
- *producto_externo*(valor_real, velocidad)
- *adición*(velocidad, velocidad)
- *desplazamiento*(posición, velocidad).

Una de las primeras aplicaciones de este modelo fue al problema del agente viajero [Zhi et al. 2004].

3. Enfoques de Optimización Multiobjetivo

Un enfoque para la optimización comúnmente usado es tomar uno de los atributos del sistema como función objetivo, y mediante el mismo determinar el orden (total) de preferencia de las soluciones factibles. Este tipo de problemas de optimización se refieren como mono-objetivo. Los demás atributos, si los hubiere, se modelan de otra forma, por ejemplo, como restricciones.

Sin embargo, existe otro enfoque, más general, que es la optimización multi-objetivo, donde varios atributos se emplean como funciones objetivo y se usan para definir un orden de preferencia parcial de las soluciones factibles. Por caso, en el problema NRP bi-objetivo presentado, se desea minimizar el costo de desarrollo y maximizar la satisfacción de los clientes. Estos objetivos, al igual que en muchos otros problemas, son de cierta forma contradictorios y compiten entre sí, definiendo sobre el espacio de soluciones un orden parcial, la relación de dominancia, donde existen pares de soluciones que no son comparables *a priori*. Ante esta situación pueden tomarse básicamente dos estrategias para la resolución del problema de optimización multi-objetivo:

(a) buscar el frente de Pareto completo, compuesto por todas las soluciones no dominadas y dejar al tomador de decisiones la tarea de buscar en ese frente la solución que mejor se adecúe a su criterio (por caso, mediante otro algoritmo), o bien
(b) utilizar algún mecanismo (por caso, una nueva relación de orden total) que reduzca el frente de Pareto a una única solución o a una zona de interés.

Un algoritmo derivado del algoritmo PSO original que sigue el enfoque (a) es el **MOPSO** (*Multi-Objective PSO*). Este algoritmo posee un repositorio finito de soluciones no dominadas, y las partículas actualizan este repositorio con el correr de las iteraciones, según las posiciones que visitan. Múltiples variantes se han formulado, y una revisión de los mismos se puede encontrar en [Zhou et al. 2011].

Por otra parte, la familia de **X-FPSO** formas está compuesta por variantes del algoritmo PSO original que utilizan una extensión *a priori* al dominio multi-objetivo mediante la utilización de la Teoría de Conjuntos Difusos. En este enfoque se utilizan las preferencias del tomador de decisiones respecto de la importancia relativa de los objetivos para componer una única función objetivo a optimizar. Cada objetivo (o restricción) es modelado mediante un número difuso cuya función de pertenencia es lineal. Esta función de pertenencia es afectada por un ponderador que la dilata o contrae, y por lo tanto, disminuye o aumenta, respectivamente, la importancia del objetivo en la decisión. Finalmente, las funciones de pertenencia ponderadas confluyen según un operador llamado t-norma, definida para lograr un único valor cuantitativo de

satisfacción de todos los objetivos simultáneamente. Como puede inferirse, este tipo de algoritmos busca una única solución, ya que la importancia de los objetivos se supone definida. Los pormenores de este enfoque se encuentran más allá del alcance de este artículo, y los detalles más importantes se pueden encontrar en la referencia [Camargo et al. 2018].

4. *Fuzzy Multi-Objective PSO*

Un aporte importante de este trabajo es la definición de un nuevo algoritmo derivado del PSO, el FMOPSO (*Fuzzy Multi-Objective PSO*), que mejora la performance de otros algoritmos del estado del arte como MOPSO en la aproximación del frente Pareto-óptimo. En esta sección se presentan los detalles de diseño del algoritmo, y de su aplicación al problema NRP bi-objetivo, dejando los resultados para la sección correspondiente.

4.1. Descripción general

El algoritmo propuesto se compone de un enjambre de partículas, dispuesto en dos dimensiones, m × n, por lo tanto, las partículas se denotan como p_{ij}. Cada partícula p_{ij} posee su posición X_{ij} y su velocidad V_{ij}. Sin embargo, la mejor posición autobiográfica, b, es compartida por las partículas en la misma columna, es decir, hay un solo b_j para cada partícula p_{ij}, con $i \in \{1, \dots, m\}$. Por lo tanto, hay n mejores posiciones autobiográficas, que pasan a ser **grupales** en lugar de individuales.

La representación de las soluciones es binaria: para cada requerimiento k, un $x_k = 1$ representa la decisión de incluir el requerimiento k en el próximo *release*. Los detalles respecto a la representación de las posiciones y velocidades se describen en detalle en la sección siguiente.

Un párrafo aparte merece la función de aptitud utilizada. Como ya se dijo, los dos objetivos de esta versión del problema NRP son el costo C de implementación, y el beneficio B dado por la satisfacción de los clientes interesados. Se compone una función de pertenencia difusa por cada objetivo, de la misma forma que en el enfoque de las X-FPSO formas mencionado en la sección anterior. Es decir que la función de pertenencia del número difuso Costo está dada por la siguiente expresión:

$$\mu_C(X, \rho_C) = \begin{cases} 1 & si \quad C(X) \leq C_{min} \\ \left(\frac{C(X) - C_{max}}{C_{min} - C_{max}}\right)^{\rho_C} & si \quad C_{min} < C(X) < C_{max} \\ 0 & si \quad C(X) \geq C_{max} \end{cases} \tag{5}$$

siendo $C(X)$ el costo de la solución X, $C_{\min}$ y $C_{\max}$ valores de referencia mínimos y máximos para la variable costo, y ρ_C un ponderador determinado previamente.

De la misma manera, la función de pertenencia del número difuso Beneficio está dada por la siguiente expresión:

$$\mu_B(X, \rho_B) = \begin{cases} 1 & si \quad B(X) \geq B_{max} \\ \left(\frac{B(X) - B_{min}}{B_{max} - B_{min}}\right)^{\rho_B} & si \quad B_{min} < B(X) < B_{max} \\ 0 & si \quad B(X) \leq B_{min} \end{cases} \tag{6}$$

siendo $B(X)$ el beneficio de la solución X, $B_{\min}$ y $B_{\max}$ valores de referencia mínimos y máximos para la variable beneficio, y ρ_B un ponderador definido previamente.

Por otra parte, este es un algoritmo que busca aproximar un frente Pareto-óptimo. Son precisamente las mejores posiciones grupales las que constituirán el frente aproximado. Sin embargo, si todas las partículas comparten la misma función objetivo, estas convergerían hacia una misma posición. Es por eso que distintas partículas se especializan en buscar en una parte distinta del frente. Cada partícula tendrá una función objetivo que depende de su posición en el arreglo bidimensional de partículas. En particular, según su posición, cada objetivo será ponderado de forma diferente, aumentando o disminuyendo la importancia de los objetivos, según corresponda. De esta manera, la función de aptitud para una posición X correspondiente a una partícula ubicada en la columna j está dada por la expresión:

$$\mu_D(X,j) = min\left\{\mu_C\left(X, \frac{2(j-1)}{n-1}\right), \mu_B\left(X, 2 - \frac{2(j-1)}{n-1}\right)\right\} \quad (7)$$

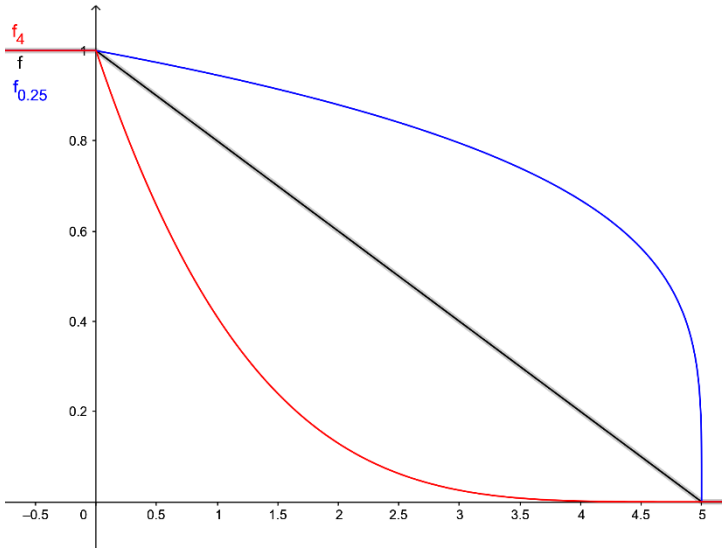


Figura 2. Conjunto difuso lineal con ponderadores 0.25 (azul), 1 (negro) y 4 (rojo)

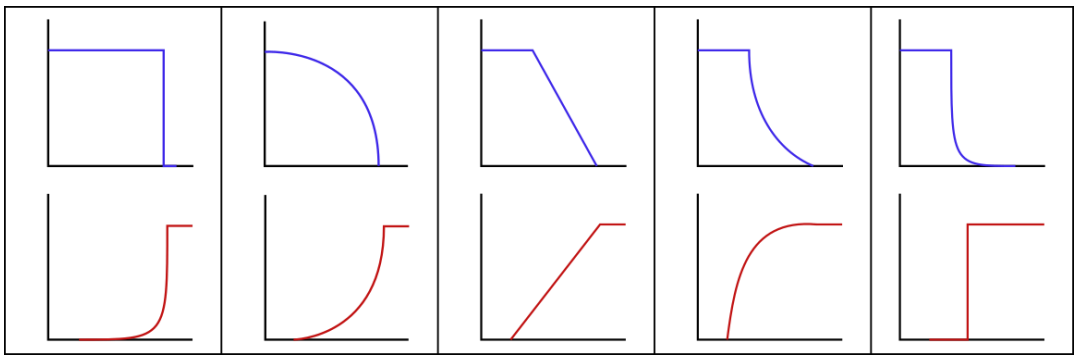


Figura 3. Ejemplo de ponderaciones de Costo (azul) y Beneficio (rojo) en una topología de 5 columnas

De esta manera, las partículas que se encuentran en las columnas de menor índice poseen una ponderación que prioriza el beneficio por sobre el costo, es decir, $\rho_B > \rho_C$. En particular, en la primera columna $\rho_B = 2$ y $\rho_C = 0$. Análogamente, las partículas en las columnas de mayor índice priorizan el costo por sobre el beneficio, esto es, $\rho_B < \rho_C$, y en particular, en la última columna, $\rho_B = 0$ y $\rho_C = 2$. La Figura 3 presenta esquemáticamente las ponderaciones de Costo (azul) y Beneficio (rojo) en una topología de 5 columnas.

4.2. Especificación del PSO Clásico

La representación de posición y velocidad para este problema se realiza de la misma forma que en el trabajo [Afshinmanesh et al. 2005], es decir, mediante vectores binarios. El tamaño de ambos vectores es igual a la cantidad de requerimientos del problema NRP. Para el caso de la posición, cada componente x_j representa la decisión de seleccionar (1) o no (0) el requerimiento j para el próximo release. Para la velocidad, un valor 1 en la posición v_j significa que se debe cambiar el valor de x_j a su complemento.

Respecto de las operaciones binarias, las mismas se definen en la Tabla 1.

Tabla 1. Especificación del modelo PSO Clásico para el FMOPSO

Operación abstracta	*resta*(X_1, X_2)	*prod_externo*($r_{1,2}$,V)	*adición*(V_1, V_2)	*desplaz*(X, V)
Operación específica	*xor*(X_1, X_2)	*and*($r_{1,2}$,V)	*or*(V_1, V_2)	*xor**(X, V)

xor, *and* y *or* representan a las operaciones binarias del álgebra de Boole ⊕, ·, +, componente a componente.

La ecuación de movimiento con este enfoque prescinde del término de inercia, resultando en consecuencia:

$$V_{ij}^{[k+1]} = r_1^{[k+1]} \cdot \left[b_j^{[k]} \oplus X_{ij}^{[k]} \right] + r_2^{[k+1]} \cdot \left[b_{Vj}^{[k]} \oplus X_{ij}^{[k]} \right] \qquad (8)$$

Naturalmente con esta definición de las operaciones, eventualmente una partícula podría caer en una posición no factible, por caso, que la solución correspondiente seleccione requerimientos que no tienen todas sus dependencias cumplidas. Por esta razón, el *xor* correspondiente al desplazamiento de las partículas es realizado en orden creciente, agregando los predecesores o quitando los sucesores, según corresponda, ante cada cambio en una componente específica.

4.3. Topología y actualización de mejores grupales

La topología de comunicación propuesta asigna tres vecinos a cada partícula, sus vecinos izquierdo y derecho, y la mejor posición grupal de la columna. Dada una partícula p_{ij}, el término de cooperación es calculado utilizando la mejor posición grupal entre sus vecinos: b_{j-1}, b_j y b_{j+1}, a excepción, claro está, de las partículas p_{i1} y p_{im}, las cuales no tienen, respectivamente, vecino izquierdo y vecino derecho.

$$\left(b_{Vj}^{[k]}, col \right) = arg \max_{l \in \{max(1, j-1), \ldots, min(j+1, n)\}} \left\{ \mu_D \left(b_l^{[k]}, l \right) \right\} \qquad (9)$$

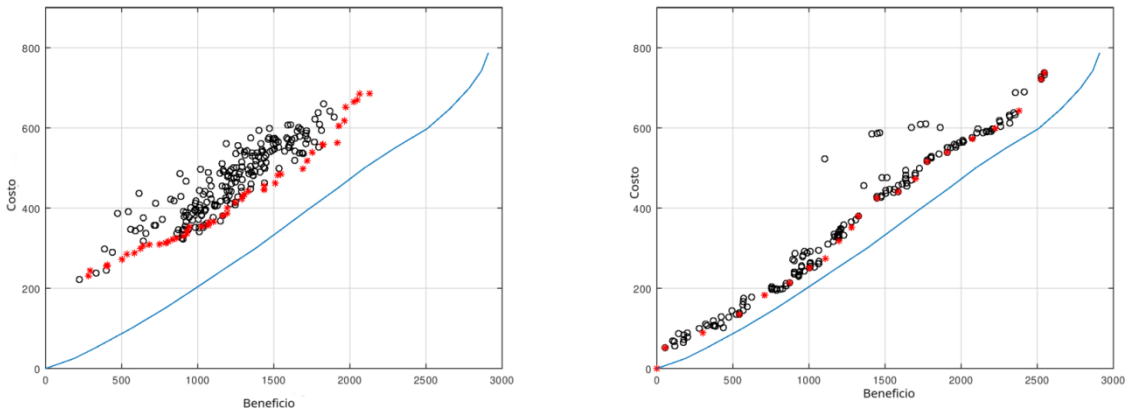
Una regla adicional se encuentra en la actualización de las mejores posiciones grupales b_j. Ante cada nueva posición $X_{ij}^{[k+1]}$, ésta es evaluada según (a lo sumo) tres versiones de la función de aptitud: $\mu_D\left(X_{ij}^{[k+1]}, j-1\right)$, $\mu_D\left(X_{ij}^{[k+1]}, j\right)$ y $\mu_D\left(X_{ij}^{[k+1]}, j+1\right)$. Ya que cada partícula es influida en su movimiento por sus vecinos izquierdo y derecho, esta puede encontrar una muy buena solución para su propia versión de la función de aptitud, o para la versión de alguno de sus vecinos. Quiere decir que al momento de actualizar las mejores posiciones grupales, b_j, las nuevas posiciones a evaluar son las $X_{il}^{[k+1]}$, con $i \in \{1, \ldots, m\}$ y $l \in \{max(1, j-1), \ldots, min(j+1, n)\}$.

5. Prueba de concepto

Se utilizó, a los efectos de probar el enfoque, una instancia del problema NRP bi-objetivo. La misma es la instancia llamada nrp1 de las instancias utilizadas en la referencia [Xuan et al. 2012] del grupo *classic*. Esta instancia tiene 140 requerimientos y 100 clientes, y contempla relaciones de precedencia entre los requerimientos.

El algoritmo FMOPSO descripto fue implementado en el lenguaje Octave, y fue comparado con una versión del algoritmo MOPSO, implementada en el mismo lenguaje, que fue adaptada por los autores para resolver el problema NRP. La versión del MOPSO original utilizada puede encontrarse en la web [Yarpiz Project 2018], y el código utilizado en este trabajo puede encontrarse en [Repositorio de código 2018].

La comparación se realizó sobre la misma instancia utilizando idéntica cantidad de partículas de movimiento. El arreglo utilizado para el FMOPSO fue de 10 × 20. Los valores de referencia de los conjuntos difusos fueron hallados mediante una optimización mono-objetivo para el Beneficio, resultando $B_{\max} = 2909$ y $C_{\max} = 787$, y se utilizó 0 tanto para $B_{\min}$ como para $C_{\min}$. Ambos algoritmos tuvieron un tiempo de ejecución de 3 minutos y 30 segundos.



(a) MOPSO (b) FMOPSO

Figura 4. Aproximación del frente Pareto- óptimo de la instancia NRP bi-objetivo

Los resultados de dos ejecuciones típicas se muestran en la Figura 4. En color rojo se encuentran los mejores grupales, en negro las posiciones de las partículas de la última iteración. A los fines de hacer más clara la comparación, se muestreó el frente Pareto-óptimo utilizando el enfoque de ε-restricciones con Programación Entera, y es el que se incluye en color azul en ambas gráficas.

La performance de los algoritmos multi-objetivo pueden compararse según criterios de convergencia, es decir, evaluando cuán lejos del frente de Pareto real están las soluciones obtenidas. En este sentido, puede verse en las figuras que el método aquí presentado proporciona soluciones que tienden a estar equiespaciadas entre sí, cubriendo una mayor parte del frente, especialmente en los extremos, y acercándose más al frente real que las proporcionadas por el algoritmo MOPSO.

Finalmente, para ilustrar el funcionamiento del algoritmo los autores han editado un video donde se muestra en tiempo real una comparación entre FMOPSO y MOPSO. El mismo puede encontrarse en [Video FMOPSO vs MOPSO 2018].

6. Conclusiones

En el presente trabajo se ha definido el problema NRP en sus versiones mono y bi-objetivo, utilizando una representación basada en Programación Lineal Entera. Además, se han presentado los aspectos teóricos relativos a un algoritmo novedoso basado en Enjambres de Partículas que permite hallar el frente Pareto-óptimo del problema NRP bi-objetivo. El algoritmo presentado fue aplicado a una instancia representante del problema NRP bi-objetivo, y se comparó la aproximación lograda con la aproximación generada por otro algoritmo del estado del arte: el MOPSO. Visualmente se pudo comprobar que el método propuesto en este trabajo logra una aproximación más cercana del frente, cubriendo una mayor porción de éste, especialmente en los extremos.

Queda aún por probar el método en otras instancias y compararlo con otros algoritmos del estado del arte. Además, todavía debe adecuarse el algoritmo para extender su alcance de modo que contemple la aproximación del frente Pareto-óptimo de problemas con más de dos objetivos.

7. Referencias

Afshinmanesh, F., Marandi, A., and Rahimi-Kian, A. (2005). A novel binary particle swarm optimization method using artificial immune system. In *Computer as a Tool, 2005. EUROCON 2005. The International Conference on*, pages 217–220. IEEE.

Bagnall, A. J., Rayward-Smith, V. J., and Whittley, I. M. (2001). The next release problem. *Information and Software Technology*.

Camargo, F., Schweickardt, G., and Casanova, C. (2018). Maps of Intrinsic Cost (IC) in reliability problems of medium voltage power distribution systems through a Fuzzy
multi-objective model. *Revista DYNA*, 85(204):334–343.

Clerc, M. (2006). *Particle Swarm Optimization*.

Clerc, M. (2013). Cooperation Mechanisms in Particle Swarm Optimisation. Technical Report 1.

Durillo, J. J., Zhang, Y., Alba, E., Harman, M., and Nebro, A. J. (2011). A study of the bi-objective next release problem. *Empirical Software Engineering*, 16(1):29–60.

Durillo, J. J., Zhang, Y., Alba, E., and Nebro, A. J. (2009). A study of the multi-objective
next release problem. In *Search Based Software Engineering, 2009 1st International Symposium on*, pages 49–58. IEEE.

Jiang, H., Zhang, J., Xuan, J., Ren, Z., and Hu, Y. (2010). A hybrid aco algorithm for the
next release problem. In *Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on*, pages 166–171. IEEE.

Kennedy, J. and Mendes, R. (2003). Neighborhood topologies in fully-informed and bestof-neighborhood particle swarms. In *SMCia 2003 - Proceedings of the 2003 IEEE
International Workshop on Soft Computing in Industrial Applications*, pages 45–50.

Repositorio de codigo (10 de Diciembre de 2018). ´ `http://github.com/casanovac/FMOPSO-Octave`.

Schweickardt, G., Gimenez Alvarez, J. M., and Casanova, C. (2016). Metaheuristics approaches to solve combinatorial optimization problems in distribution power systems.
An application to Phase Balancing in low voltage three-phase networks. *International
Journal of Electrical Power and Energy Systems*, 76:1–10.

Veerapen, N., Ochoa, G., Harman, M., and Burke, E. K. (2015). An Integer Linear Programming approach to the single and bi-objective Next Release Problem. *Information and Software Technology*.

Video FMOPSO vs MOPSO (10 de Diciembre de 2018). `http://youtu.be/DpfFECT8KF4.`

Xuan, J., Jiang, H., Ren, Z., and Luo, Z. (2012). Solving the Large Scale Next Release Problem with a Backbone-Based Multilevel Algorithm. *IEEE Transactions on Software Engineering*, 38(5):1195–1212.

Yarpiz Project (22 de Agosto de 2018). `http://yarpiz.com/59/ypea121-mopso.`

Zhang, Y., Harman, M., and Mansouri, S. A. (2007). The multi-objective next release problem. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation - GECCO '07*, page 1129. ACM Press.

Zhi, X., Xing, X., Wang, Q., Zhang, L., Yang, X., Zhou, C., and Liang, Y. (2004). A discrete PSO method for generalized TSP problem. *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*,
4(August):26–29.

Zhou, A., Qu, B. Y., Li, H., Zhao, S. Z., Suganthan, P. N., and Zhangd, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49.