






# Diseño automático de cerchas de gran escala: una comparación entre algoritmos libres de derivadas

 Luis Niño-Alvarez<sup>1</sup>,  Jeffrey Guevara-Corzo<sup>2</sup>,  Oscar Begambre-Carrillo<sup>3</sup>

Recepción: 29-04-2020 | Aceptación: 24-08-2020 | En línea: 11-11-2020

MSC: 90C59

doi:10.17230/ingciencia.16.32.4

---

## Resumen

El diseño de estructuras metálicas tipo cercha es un problema frecuente en la ingeniería civil, que requiere de la experiencia del ingeniero diseñador para lograr una solución estructural con buen desempeño y que pueda satisfacer las necesidades establecidas. En los últimos años, el diseño de estos sistemas ha sido soportado mediante la aplicación de diversos métodos de optimización, que permiten obtener soluciones óptimas, dando cumplimiento a los objetivos de diseño propuestos, de forma automática y en un menor tiempo de trabajo. Esta investigación presenta la aplicación de una serie de algoritmos metaheurísticos multiobjetivo para el diseño automático de cerchas de gran escala. Se aplicaron los algoritmos NSGA-II, MOPSO y AMOSA, y se consideraron estructuras

---

<sup>1</sup> Universidad Industrial de Santander, luis2188256@correo.uis.edu.co, Bucaramanga, Colombia.

<sup>2</sup> Universidad Industrial de Santander, jeffrey.guevara1@correo.uis.edu.co, Bucaramanga, Colombia.

<sup>3</sup> Universidad Universidad Industrial de Santander, ojbegam@uis.edu.co, Bucaramanga, Colombia.

reportadas en la literatura conformadas por un número elevado de elementos. El desempeño de los algoritmos se evaluó con base en el costo computacional, el criterio del hipervolumen y el comportamiento que tienen los algoritmos al aumentar la cantidad de iteraciones por ciclo de optimización. El espacio de búsqueda usado en la optimización fue discreto, restringido por los perfiles de acero W disponibles en el mercado colombiano. Los resultados obtenidos demuestran que, para los problemas propuestos, el algoritmo MOPSO es el más eficiente, seguido del AMOSA y del NSGA-II que mostró un costo computacional mayor. Finalmente, vale la pena mencionar que los tiempos de cálculo fueron menores a una hora, para cerchas cercanas a los mil elementos.

**Palabras clave:** Optimización metaheurística multiobjetivo; estructuras articuladas; cercha; gran escala.

---

## Automatic Design of Large-Scale Trusses: A Comparison Between Derivative-Free Algorithms

---

### Abstract

The design of steel trusses is a frequent problem in civil engineering, which requires the experience of the design engineer to achieve a structural solution with good performance and that can satisfy the established needs. In recent years, the design of these systems has been supported by the application of various methods of optimization, which allow optimal solutions, meeting the proposed design objectives, automatically and in a shorter time. This research presents the application of a series of multi-objective metaheuristic algorithms for the automatic design of large-scale trusses. The NSGA-II, MOPSO and AMOSA algorithms were applied and the structures reported in the literature were considered to be made up of a high number of elements. The performance of the algorithms was evaluated based on the computational cost, the hypervolume criterion and the behavior that the algorithms have when increasing the amount of iterations per optimization cycle. The search space used in the optimization was discrete, restricted by the W steel profiles available in the Colombian market. The results obtained show that, for the proposed problems, the MOPSO algorithm is the most efficient, followed by the AMOSA and the NSGA-II which showed a higher computational cost. Finally, it is worth mentioning that the calculation times were less than one hour, for trusses close to a thousand elements.

**Keywords:** Multi-objective metaheuristic optimization; articulated structures; trusses; large scale.

---

## 1 Introducción

La optimización estructural de cerchas se ha constituido en un tema de gran interés por parte de la comunidad científica, ya que su uso proporciona soluciones de diseño preliminares que puede perfeccionarse con análisis adicionales para su implementación en el mundo real [1], lo que permite obtener soluciones estructurales cuasi-óptimas.

El diseño de estructuras metálicas tipo cercha, es un proceso iterativo en el cual se prueban diferentes opciones (ej. diversas combinaciones de perfiles estructurales [2],[3]) hasta dar cumplimiento a las normas de diseño. Lo anterior se logra generalmente con un número limitado de pruebas, dada la alta cantidad de cálculos que el diseñador debe realizar. La aplicación de optimización estructural basada en los algoritmos metaheurísticos, permite que este proceso de diseño pueda llevarse a cabo de manera automática e iterativa, realizando una alta cantidad de pruebas en tiempos razonables, proporcionando soluciones óptimas satisfactorias [2],[3],[4].

Por otro lado, cuando se aplica un proceso de optimización estructural en cerchas, existen tres tipos de optimización de acuerdo a las variables de diseño que son; la optimización de tamaño, donde las variables de diseño son las áreas de los elementos, manteniendo fija la conectividad de los elementos y las coordenadas de los nodos; seguido de la optimización de forma o geométrica, en el que las variables de diseño son las coordenadas de los nodos, manteniendo fija la conectividad de los elementos y el área de los elementos; y finalmente la optimización topológica discreta, en el que las variables de diseño son las áreas de los elementos y se busca determinar la conectividad de los elementos, manteniendo fijo las coordenadas de los nodos (pueden existir nodos inactivos) [5].

En los últimos años, se han aplicado una gran variedad de algoritmos metaheurísticos en el problema de optimización de cerchas [6],[7],[8], los cuales han presentado buenos resultados y han establecido estrategias efectivas para resolver problemas de optimización como las presentadas por Assimi *et al.* [9],[1], Goncalves *et al.*[10], Miguel *et al.*[11] entre muchos otros. Sin embargo, los trabajos que tratan con problemas de optimización de estructuras tipo cercha de gran escala (cientos o miles de elementos [12]) son limitados [13],[14],[15],[16], en especial los que consideran múltiples objetivos en su proceso de optimización[6],[17],[18], ya que la mayor parte

de los estudios presentan aplicaciones de los algoritmos metaheurísticos en problemas de estructuras de pequeña escala y con una sola función objetivo, generalmente el peso estructural.

En este artículo, se presentan los resultados de la aplicación de los algoritmos metaheurísticos multiobjetivo NSGA-II [19], MOPSO [20],[21] y AMOSA [22] en el diseño óptimo automático de 4 estructuras tipo cercha de gran escala de 120, 200, 582 y 942 elementos respectivamente, las cuales han sido estudiadas por autores como Kaveh *et al.* [23],[24],[25],[26], Mortazavi *et al.*[27],[28], Bekdas *et al.*[29] entre otros. Con el fin de evaluar el comportamiento de los algoritmos en estos problemas de complejidad considerable, se usaron los parámetros recomendados por los autores originales. Adicionalmente, las 4 estructuras estudiadas se encuentran sometidas a las mismas restricciones de diseño y se estableció un número de iteraciones (250, 500 y 750) en todos los ejercicios de aplicación, manteniendo los 3 en igualdad de condiciones. Sumado a lo anterior, con el fin de observar la convergencia que presenta cada algoritmo, se repitió el ejercicio de diseño aumentando el número de iteraciones por ciclo de optimización. Se consideró la minimización del peso de la estructura y la energía de deformación como funciones objetivo y finalmente, el problema se planteó con un espacio de búsqueda discreto compuesto por un grupo de 35 perfiles W.

El documento está estructurado de la siguiente forma. En la sección 2 se describe el concepto básico de la optimización multiobjetivo, posteriormente, en la sección 3 se tratan los algoritmos metaheurísticos multiobjetivo usados (NSGA-II, MOPSO y AMOSA). La sección 4 se presentan las métricas que se usaron para evaluar el desempeño, en la sección 5, se muestran las funciones objetivo, restricciones de diseño y los problemas abordados, en la sección 6 se muestran los resultados que se obtuvieron en el proceso de diseño y optimización y para finalizar, en la sección 7 se muestran las conclusiones de este estudio.

## 2 Optimización Multiobjetivo

La optimización en términos generales, es la búsqueda del valor extremo (máximo o mínimo) de una función objetivo dependiente de unas variables

y sujeto a una serie de restricciones. Muchos de los problemas complejos de la industria pueden formularse como un proceso de optimización (ej. minimizar tiempo o maximizar utilidades). Cuando el problema de optimización presenta más de una función objetivo (ej. minimizar el costo y maximizar la calidad), se tiene un problema de optimización multiobjetivo, donde la complejidad aumenta, ya que no existe una única solución óptima para las funciones objetivo consideradas, sino un conjunto de soluciones óptimas (no dominadas) llamadas soluciones Pareto-óptimas, que representadas gráficamente se conocen como frente de Pareto [30]. La generación del frente Pareto tiene múltiples ventajas, especialmente la de obtener un amplio rango de soluciones Pareto óptimas, en donde el diseñador puede elegir la que más se ajusta a las necesidades [31].

En general, un problema de optimización multiobjetivo se establece de la siguiente forma [32]:

$$\min \text{ o } \max F(x) = z = F_m(x) \quad m = 1, 2, 3, \dots, M \quad (1)$$

sujeto a.

$$g_j(x) \geq 0 \quad j = 1, 2, 3, \dots, J \quad (2)$$

$$h_k(x) = 0, \quad k = 1, 2, 3, \dots, K \quad (3)$$

$$x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, 3, \dots, n \quad (4)$$

Donde  $x = (x_1, x_2, \dots, x_n)$  es el vector de las  $n$  variables de diseño,  $F(x)$  es el vector de las  $M$  funciones objetivo,  $g_j(x)$  son las funciones de restricción de desigualdad,  $h_k(x)$  las funciones de restricciones de igualdad,  $x_i^{(L)}$  y  $x_i^{(U)}$  son las condiciones de borde o el límite de las variables de diseño  $x_i$ .

Por otro lado, el proceso de diseño de cualquier tipo de estructura, es un proceso complejo que requiere establecer una topología (patrón de conectividad de los elementos), geometría (coordenadas de los nodos) y un conjunto de perfiles que permitan dar cumplimiento a las restricciones normativas de diseño. En este sentido, el proceso de diseño estructural puede ser planteado como un problema de optimización multiobjetivo, que permita facilitar y establecer la mejor opción de diseño dentro de un conjunto de soluciones Pareto óptimas disponibles.

Para el caso de estructuras tipo cercha, los problemas de optimización multiobjetivo buscan maximizar o minimizar el vector de funciones objetivo, que representa criterios estructurales (peso, rigidez, energía de deformación, entre otras), dependientes de variables de diseño (área de los elementos) y sujetas a un número definido de restricciones (esfuerzo en los elementos, desplazamiento en los nodos, entre otras) [33].

### 3 Algoritmos metaheurísticos multiobjetivo

Estos algoritmos son métodos de optimización que, con frecuencia están orientados hacia el cálculo directo de un frente de Pareto, optimizando simultáneamente las funciones objetivo. Estos algoritmos son particularmente robustos cuando no se conoce información acerca de las preferencias o la prioridad de las funciones objetivo, y cuando se quiere un amplio rango de alternativas de solución [31],[34]. En las siguientes subsecciones, se describe de forma general cada uno de los algoritmos usados.

#### 3.1 NSGA-II(*Non-Dominated Sorting Genetic Algorithm II*)

Los algoritmos genéticos o GA, son un tipo de algoritmo evolutivo inspirado en los principios de genética natural, usados con frecuencia para resolver problemas de optimización en distintas áreas. El NSGA-II es una versión multiobjetivo del GA, propuesto por Deb *et al.* [19], siendo uno de los algoritmos metaheurísticos evolutivos multiobjetivo más usados para la resolución de problemas de optimización multiobjetivo.

#### 3.2 MOPSO(*Multiobjective Particle Swarm Optimization*)

El algoritmo PSO (*Particle Swarm Optimization*) es un algoritmo metaheurístico que en un principio fue ideado para el modelamiento del comportamiento social de las especies animales como las aves o los peces, sin embargo, posteriormente el algoritmo fue mejorado y se empezó a aplicar en problemas de optimización. En el algoritmo, un individuo miembro de

la población se conoce como partícula, y su comportamiento está afectado por su mejor ubicación local histórica ( $p_{best}$ ), y el mejor global ( $g_{best}$ ) de todas las partículas, lo cual permite que se beneficien de las experiencias pasadas.

El algoritmo MOPSO propuesto por Coello *et al.* [21],[20] se basa en la idea de tener un repositorio global, en el cual, cada partícula deposita sus experiencias después de cada iteración. Adicionalmente, la actualización del repositorio se realiza considerando un sistema geográfico, en términos del valor de las funciones objetivo de cada individuo o partícula. Esta técnica está inspirada en el archivo externo usado en el algoritmo PAES (*Pareto Archived Evolution Strategy*).

El algoritmo usa el concepto de población (partículas) y una medida de desempeño similar al valor de *fitness* en los algoritmos evolutivos [20]. La analogía del PSO con los algoritmos evolutivos, es lo que permite que el esquema de *Pareto ranking* pueda ser usado de manera sencilla para su aplicación en la solución de problemas de optimización multiobjetivo. El archivo histórico de las mejores soluciones encontradas por una partícula puede usarse para guardar las soluciones no dominadas generadas en el pasado. El uso de mecanismos de atracción global, combinado con un archivo histórico que guarde las mejores soluciones no dominadas encontradas previamente, motiva la convergencia hacia las soluciones globales no dominadas [20].

### 3.3 AMOSA (*Archived Multiobjective Simulated Annealing*)

El algoritmo SA (*Simulated Annealing*) es un algoritmo heurístico usado con popularidad en optimización, que se basa en los principios del recocido o templado de metales.

El AMOSA es una versión multiobjetivo del algoritmo SA, desarrollada por Bandyopadhyay *et al.* [22]. El algoritmo incorpora el concepto de 'Archivo', donde se guardan las mejores soluciones no dominadas encontradas durante el proceso de optimización. El tamaño del Archivo es definido, ya que solo es necesario obtener un número limitado de soluciones óptimas de Pareto bien distribuidas. Se establecen dos límites para el tamaño del archivo: un límite estricto denotado como HL (*Hard Limit*)

y un límite flexible denotado como SL (*Soft Limit*). Durante el proceso de optimización, las soluciones no dominadas se guardan en el archivo a medida que se generan, hasta que el tamaño del archivo alcanza el valor de SL, después de esto, si se generan más soluciones no dominadas, estas se añaden al archivo; como se ha superado el valor de SL, este se reduce hasta alcanzar el valor de HL usando una técnica para agrupar soluciones (*Clustering*).

## 4 Desempeño de los algoritmos

Para la evaluación del desempeño de los algoritmos en los ejercicios propuestos, se considera el costo computacional y el hipervolumen de las soluciones de los frentes de Pareto obtenidos.

Para la evaluación del costo computacional, todos los algoritmos de optimización multiobjetivo se programaron y ejecutaron en el software Matlab v2019a [35] y un equipo de computo con un procesador Intel Core i7 7700(3.6 GHz) con 16GB de RAM.

### 4.1 Hipervolumen (HV)

El hipervolumen es un parámetro que permite evaluar de forma cualitativa la convergencia y la diversidad de un algoritmo de forma combinada [32]. Este calcula el volumen cubierto por los elementos del frente de Pareto obtenido  $Q$ , en problemas donde todos los objetivos son minimizados. Matemáticamente para cada solución  $i$  que pertenece al frente  $Q$ , existe un hipervolumen  $V_i$  que se construye con base a un punto de referencia  $W$  y cada solución  $i$  como las esquinas diagonales del hipercubo, siendo el punto de referencia  $W$  los peores valores de las funciones objetivo. La unión de todos los hipercubos calculados, constituye el hipervolumen  $HV$ , calculado con la ecuación (5), donde el mejor valor de  $HV$  es 1 (para funciones objetivo normalizadas).

$$HV = Volumen \left( \bigcup_{i=1}^{\|Q\|} V_i \right) \quad (5)$$



## 5 Problema de optimización

Para evaluar el comportamiento y desempeño de los algoritmos, se hicieron 5 ciclos de optimización con 250, 500 y 750 iteraciones por ciclo de optimización para cada ejercicio. Para esto, se escogieron 4 estructuras tipo benchmark estudiadas por autores como Kaveh *et al.* [23],[25], Cheng *et al.* [36], Degertekin *et al.* [16] y Hasańcebi [2], entre otros, con 120, 200, 582 y 942 elementos, considerando como funciones objetivo el peso de la estructura (W) y energía de deformación (E) (ver Ecuaciones (6) y (7)).

$$W = \rho \sum_{i=1}^n A_i L_i \tag{6}$$

$$E = \frac{1}{2} \sum_{i=1}^n F_i \delta_i \tag{7}$$

A diferencia de lo desarrollado por los distintos autores, en este caso, los 4 ejercicios de optimización se desarrollaron en igualdad de condiciones. Las propiedades de los materiales, restricciones y el número de variables de diseño para las 4 estructuras son las siguientes.

**Tabla 1:** Características de los problemas de optimización

Propiedades del material	$\rho = 0,228 \text{ lb/in}^3$ $E = 30450 \text{ ksi}$ $F_y = 58 \text{ ksi}$
Restricciones	$\sigma_{lim}^+ = 0,6 F_y$ $\sigma_{lim}^- = \frac{\left(1 - \frac{\lambda^2}{2C_c}\right) F_y}{\frac{5}{8} + \frac{3\lambda}{8C_c} - \frac{\lambda^3}{8C_c^3}} \text{ for } \lambda < C_c$ $\sigma_{lim}^- = \frac{12\pi^2 E}{23\lambda^2} \text{ for } \lambda > C_c$ $\lambda = KL/r$ $\lambda_{lim} \rightarrow K_m L_m / r_m < 300 \text{ para } F^+$ $\lambda_{lim} \rightarrow K_m L_m / r_m < 200 \text{ para } F^-$ $C_c = \sqrt{2\pi^2 E / F_y}$
Variables de diseño	7 Grupos de Variables - Armadura de 120 Elementos 29 Grupos de Variables - Armadura de 200 Elementos 32 Grupos de Variables - Armadura de 582 Elementos 59 Grupos de Variables - Armadura de 942 Elementos

Los parámetros de cada algoritmo de optimización multiobjetivo se presentan en las Tablas 2, 3 y 4 .

**Tabla 2:** Tabla de parámetros NSGA-II

Parámetro	Valor
Num. Máximo Ite.	250, 500 , 750
Tamaño Pob.	100
Ind. Dist. SBX	20
Ind. Dist. PBMO	20
Prob. Cruzamiento	0.9
Prob. Mutación	1/(Num.Var.)
Num. Eval. / Ciclo	25000, 50000 y 75000

**Tabla 3:** Tabla de parámetros MOPSO

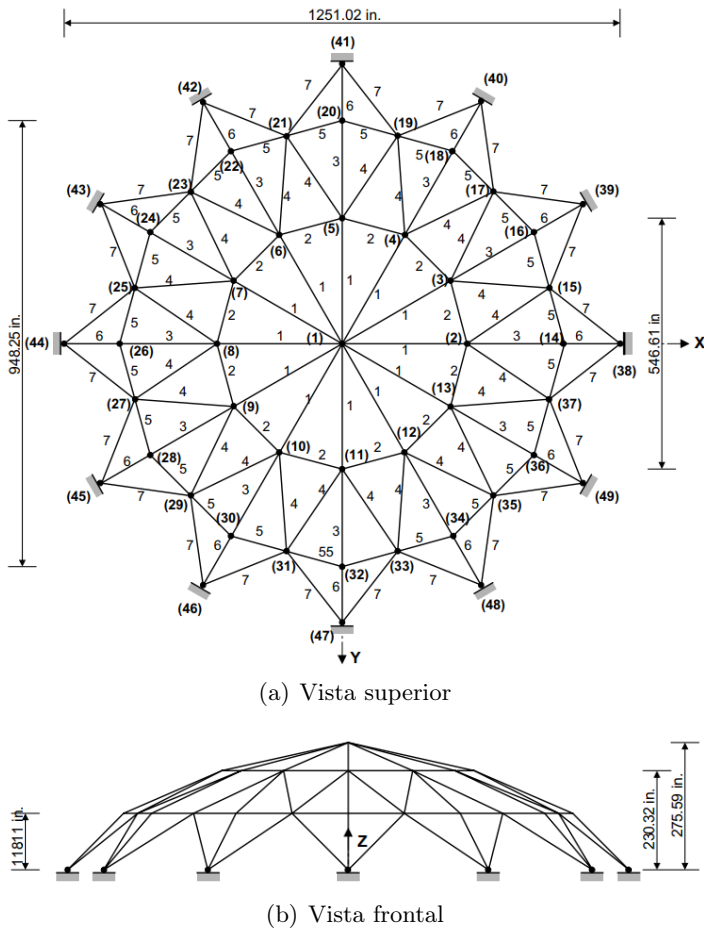
Parámetro	Valor
Num. Máximo Ite.	250, 500 , 750
Num. Partículas	100
Tamaño Rep.	100
Tamaño Div. Rep.	7
Coef. Acel. Cog.	2
Coef. Acel. Soc.	2
Op. de Mutación	0.1
In. con Dec. Lineal	Wmax=0.9 y Wmin=0.4
Num. Eval. / Ciclo	25000, 50000 y 75000

**Tabla 4:** Tabla de parámetros AMOSA

Parámetro	Valor
Num. Máximo Ite.	250, 500 , 750
Num. Ite. / Temp	100
Num. Ite. Hill-Climb.	10
Temp. Máxima	200
Temp. Mínima	1.00E-06
Tamaño Max. HL	100
Tamaño Max. SL	150
Parametro Gamma	2
Dec. Temp.	0.8
Num. Eval. / Ciclo	25000, 50000 y 75000

### 5.1 Armadura de 120 elementos

En este primer problema, estudiado por Fallahian *et al.* [37] y otros, se desarrolla la optimización de una armadura de 120 elementos conectados mediante 49 nodos, como se muestra en la Figura 1.

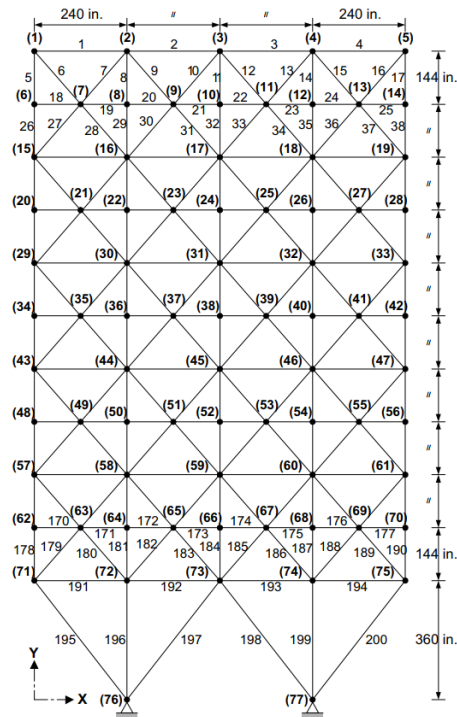


**Figura 1:** Armadura de 120 elementos. - **Fuente:** Lee *et al.* [38]

Sumado a las restricciones presentadas en la Tabla 1, el problema tiene una restricción en los desplazamientos y el área de los elementos estructurales, limitando los nodos de la estructura a un desplazamiento máximo de 0.1969 inch y un área mínima de un elemento de 0.775 inch<sup>2</sup>. La estructura está sujeta a cargas verticales en los nodos no soportados de -13.41 kips (Z) en el nodo 1, -6.744 kips (Z) en los nodos 2 al 13, y -2.248 kips (Z) en los nodos restantes.

## 5.2 Armadura de 200 elementos

En este segundo problema, trabajado por Cheng *et al.* [36] y otros, se desarrolla la optimización de una armadura de 200 elementos conectados mediante 77 nodos, como se muestra en la Figura 2.

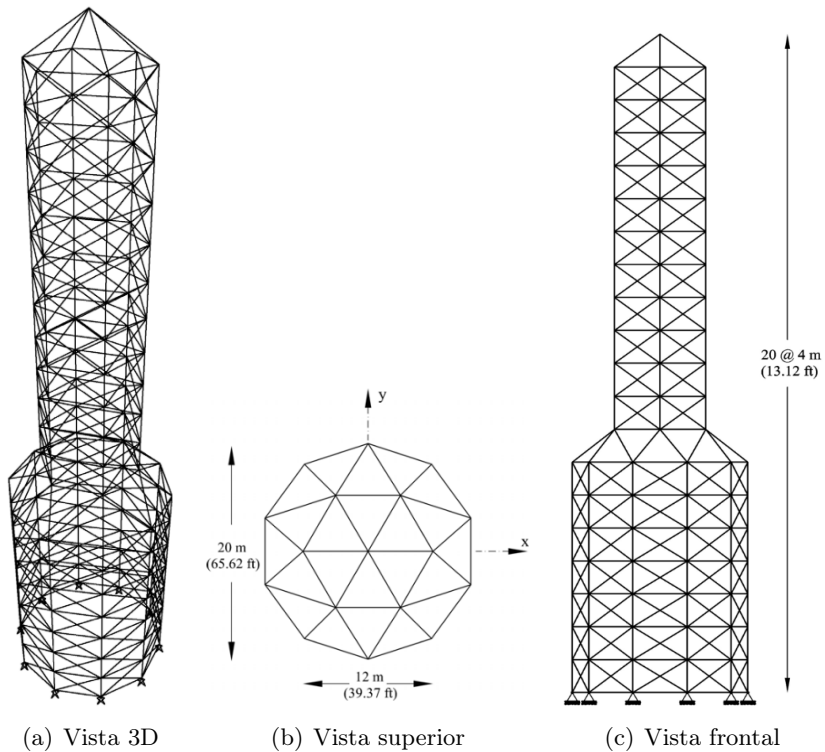


**Figura 2:** Armadura de 200 elementos. - **Fuente:** Lee *et al.* [38]

La estructura está sujeta a 3 diferentes estados de cargas. 1) 1 kip (X) en los nodos 1, 6, 15, 20, 29, 34, 43, 48, 57, 62 y 71; 2) 10 kips (Y) en los nodos 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 22, 24, 26, 28, 30, 31, 32, 33, 34, 36, 38, 40, 42, 43, 44, 45, 46, 47, 48, 50, 52, 54, 56, 57, 58, 59, 60, 61, 62, 64, 66, 68, 70, 71, 72, 73, 74 y 74; 3) la combinación de los caso de carga 1 y 2, actuando de manera conjunta.

### 5.3 Armadura de 582 elementos

En este tercer problema, trabajado por Kaveh *et al.* [23],[26] y otros, se desarrolla la optimización de una armadura de 582 elementos conectados mediante 153 nodos, como se muestra en la Figura 3.

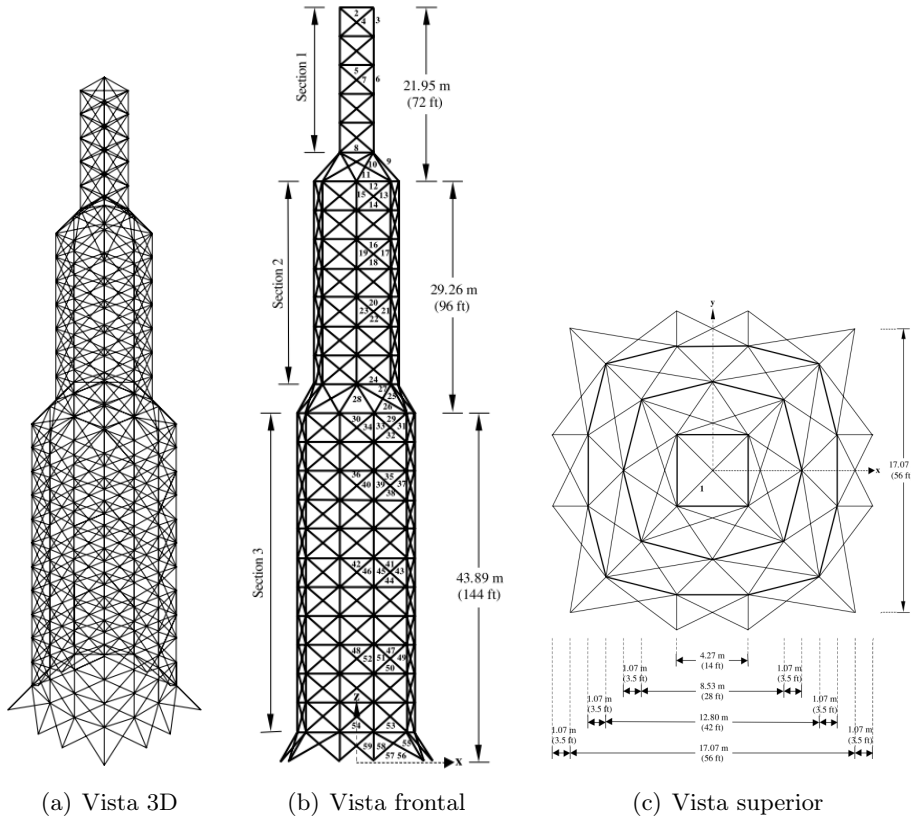


**Figura 3:** Armadura de 582 elementos. - **Fuente:** Kaveh *et al.* [23]

La estructura está sujeta a un único estado de cargas, de 5kN en dirección X, 5kN en dirección Y, -30kN en la dirección Z en todos los nodos de la estructura.

### 5.4 Armadura de 942 elementos

En este cuarto problema, trabajado por Degertekin *et al.* [16] y otros, se desarrolla la optimización de una armadura de 942 elementos conectados mediante 153 nodos, como se muestra en la Figura 4.



**Figura 4:** Armadura de 942 elementos. - Fuente: Hasançebi [2]

La estructura está sujeta a 3 diferentes estados de cargas. 1) Carga de -3 kips, -6 kips y -9 kips (Z) en los nodos de la primera, segunda y tercera sección de la estructura respectivamente; 2) Carga de 1 kip (Y) en todos los nodos de la torre; 3) Carga de 1.5 kips y 1 kip (X) en los nodos del lado izquierdo y derecho de la torre respectivamente.

## 6 Resultados

Los frentes de Pareto y los valores de HV obtenidos para los 5 ciclos de optimización para cada uno de los ejercicios se presentan agrupados de acuerdo a la cantidad de iteraciones por ciclos de optimización. Los resultados obtenidos fueron los siguientes.

### 6.1 250 iteraciones

En este ejercicio es posible apreciar (ver Figura 6), que en la estructura de 120 elementos, los 3 algoritmos de optimización fueron eficientes, obteniendo frentes de Pareto con soluciones bien distribuidas. Por otro lado, para las estructuras de 200, 582 y 942 elementos, los frentes de Pareto obtenidos comienzan a ser más dispersos haciendo notorios los siguientes detalles: el AMOSA genera frentes con dispersiones pronunciadas en algunas zonas, el MOPSO genera los frentes más cercanos a los ejes, pero no tienen un buen espaciamiento y finalmente el NSGA-II genera frentes bien distribuidos, sin embargo para la estructura de 942 elementos se hace evidente que los frentes generados por el NSGA-II se encuentran más alejados. Por lo tanto, el algoritmo con el mejor comportamiento fue el MOPSO, seguido del AMOSA y finalmente del NSGA-II.

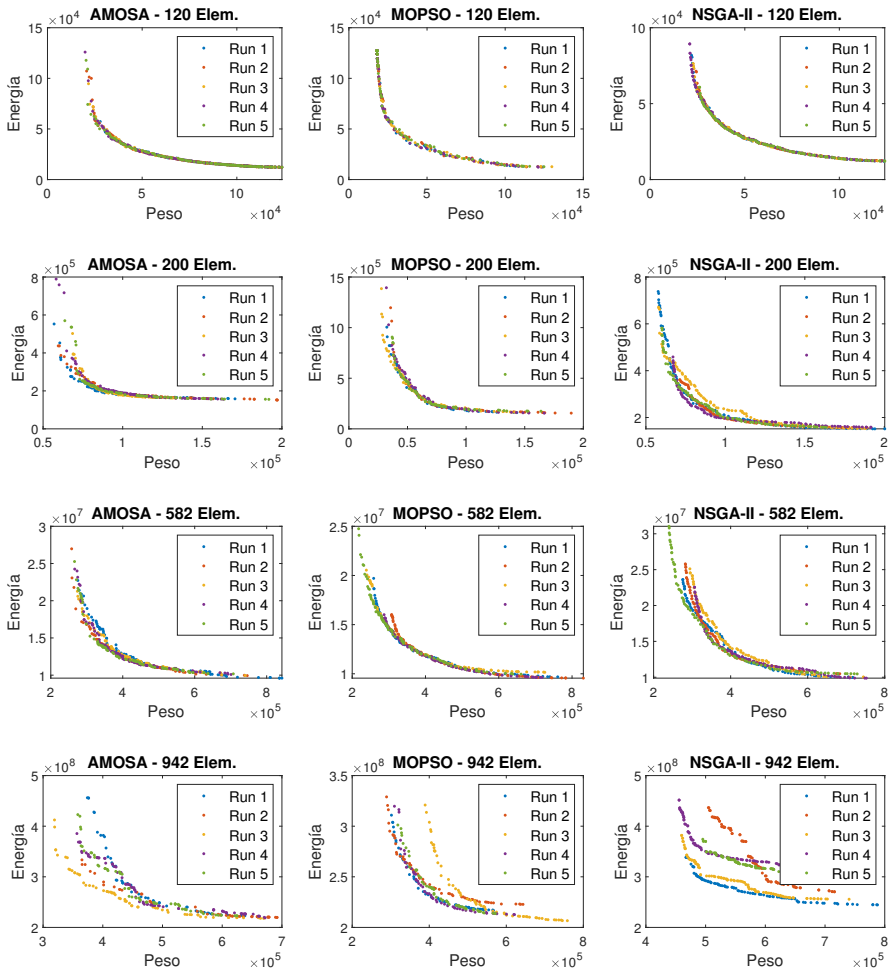


Figura 5: Frentes de Pareto para 5 ciclos de optimización de 250 iteraciones.

## 6.2 500 iteraciones

En este ejercicio como lo muestra la Figura 6, se siguen haciendo evidentes características que se aprecian en la sección anterior (250 iteraciones), sin embargo, se nota una mejora en la convergencia de los algoritmos con los frentes de Pareto obtenidos para las estructuras 582 y 942 elementos, con frentes más cercanos y mejor distribuidos, sin embargo es evidente la



dificultad que tiene el algoritmo NSGA-II en obtener el frente de Pareto, generando soluciones alejadas.

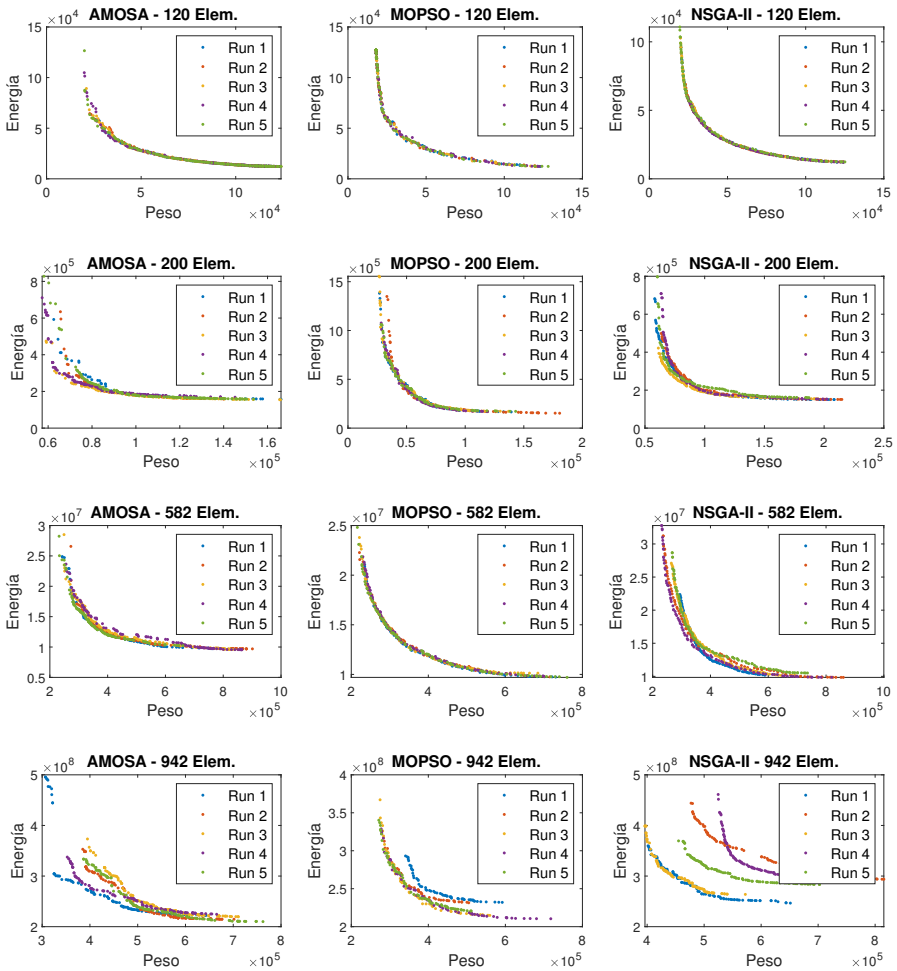


Figura 6: Frentes de Pareto para 5 ciclos de optimización de 500 iteraciones.

### 6.3 750 iteraciones

En este ejercicio como lo muestra la Figura 7, el proceso de optimización mejora de forma sustancial, obteniendo resultados favorables para las

estructuras de 120, 200 y 582 elementos. No obstante, en la estructura de 942 elementos se hace evidente que los algoritmos AMOSA y NSGA-II están generando soluciones alejadas respecto al frente de Pareto generado por el MOPSO.

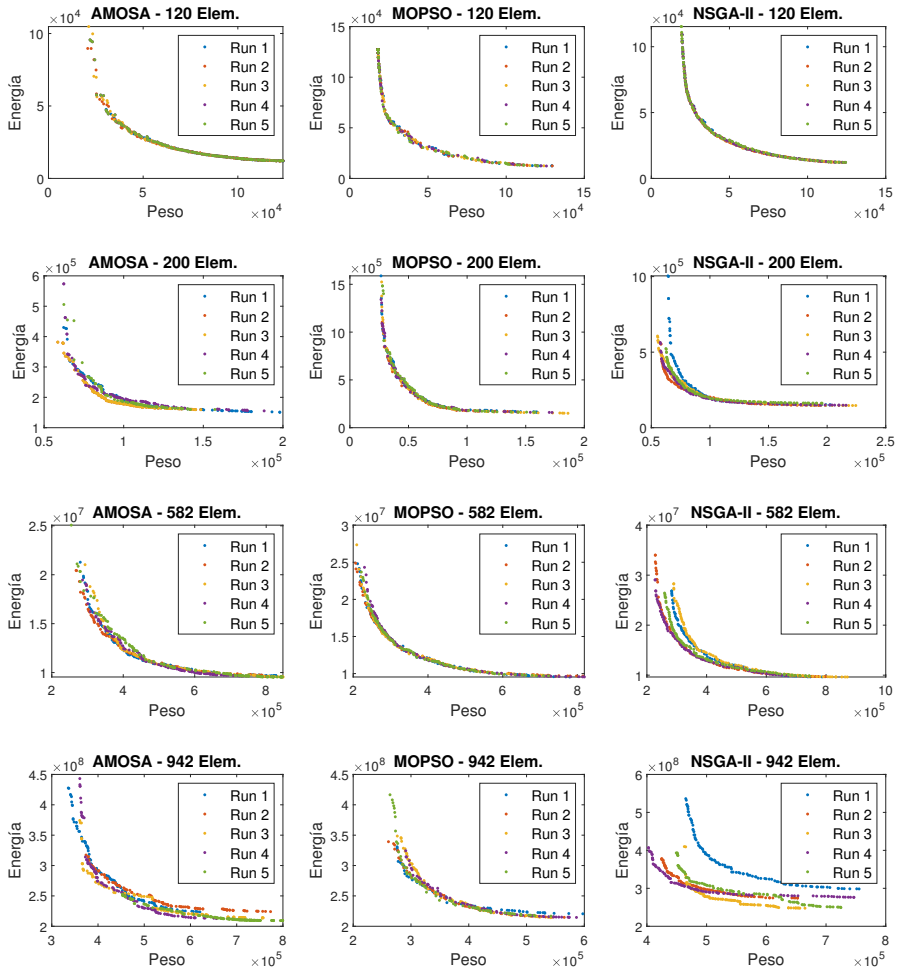


Figura 7: Frentes de Pareto para 5 ciclos de optimización de 750 iteraciones.

## 6.4 Variación y convergencia

Para obtener mayor detalle sobre la convergencia de los algoritmos, se generaron las envolventes (soluciones no dominadas de los 5 ciclos) para las estructuras estudiadas y se obtuvieron los siguientes resultados.

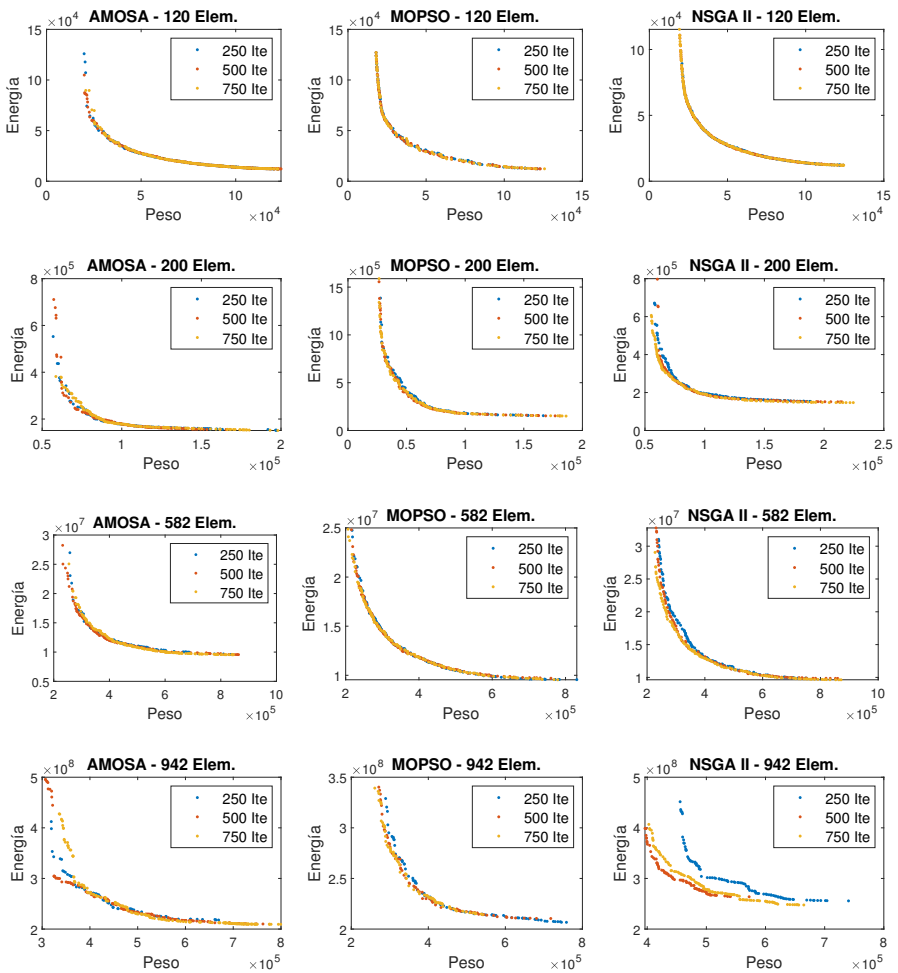


Figura 8: Envolventes de los frentes de Pareto.

Al analizar las envolventes (soluciones no dominadas de los 5 ciclos) de cada uno de los ejercicios (ver Figura 8), se hace evidente los comportamientos mencionados en secciones anteriores, donde los frentes de Pareto del algoritmo AMOSA son dispersos en algunas zonas, los frentes de Pareto del algoritmo MOPSO a pesar de tener las mejores soluciones (más cercano a los ejes), tiene problemas en la distribución generando frentes con soluciones concentradas y finalmente el algoritmo NSGA-II presenta los frentes con las soluciones mejor distribuidas, sin embargo, para la estructura de 942 elementos, el proceso de optimización requiere más iteraciones para llegar a un resultado similar al del NSGA-II y el MOPSO.

### 6.5 Costo computacional e hipervolumen

De acuerdo a los resultados del HV promedio (ver Figura 9) de los problemas propuestos, los mejores valores se obtuvieron con el algoritmo MOPSO, seguido de los algoritmos AMOSA y NSGA-II respectivamente.

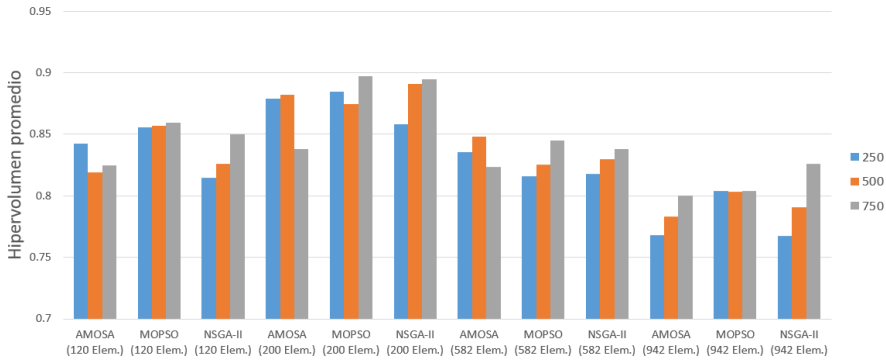
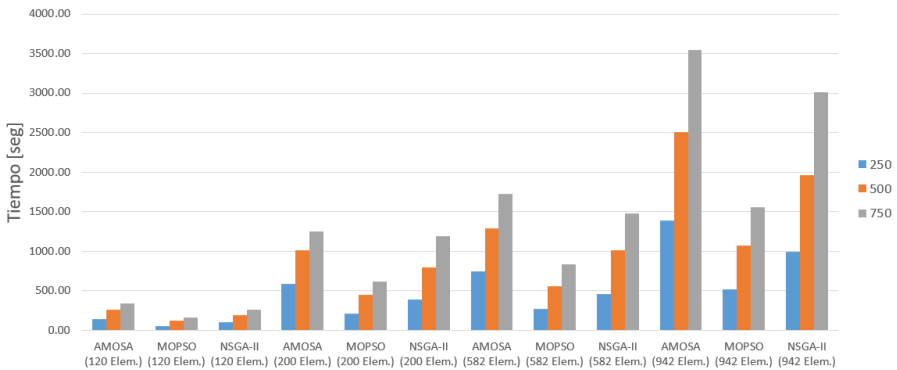


Figura 9: Hipervolumen Promedio

Adicionalmente, se aprecia un comportamiento consistente, donde al aumentar el número de iteraciones por ciclo de optimización, aumenta su HV, con excepción del algoritmo AMOSA en la estructura de 120, 200 y 582 elementos y el algoritmo MOPSO en la estructura 200 942 en donde el HV disminuyó o no varió significativamente.

Con respecto al costo computacional medido en tiempo de procesamiento (en segundos), el algoritmo más eficiente en las 4 estructuras utilizadas, fue el MOPSO, seguido por NSGA-II y finalmente AMOSA.



**Figura 10:** Costo computacional en segundos

## 7 Conclusiones

Para los 4 problemas abordados, todos los algoritmos generaron resultados prometedores manteniendo la estabilidad con costos computacionales razonables, sin embargo, en todos los casos los mejores frentes de Pareto (soluciones no dominadas) los generó el algoritmo MOPSO, seguido del AMOSA y NSGA-II respectivamente. Es necesario resaltar que los dos primeros generaron frentes con una concentración de soluciones en algunas regiones del frente de Pareto y en algunos ejercicios donde fue evidente la falta de diversidad de las soluciones, aspecto en donde el NSGA-II destacó.

En términos del costo computacional, el algoritmo más eficiente fue el MOPSO, seguido del NSGA-II y el AMOSA. En donde el MOPSO fue en promedio 2 veces más eficiente que el NSGA-II y 3 veces más eficiente que el AMOSA.

Según los resultados del Hipervolumen, al igual que con el costo computacional, el algoritmo más eficiente fue el MOPSO, siendo el que mejores resultados obtuvo en la mayor parte de los ejercicios (excepto en

la estructura de 942 elementos, 750 iteraciones), seguido del AMOSA y NSGA-II con resultados similares.

Por último, como una perspectiva futura, se podría estudiar el comportamiento que desarrollan esta clase de algoritmos, sometiéndolos a un mayor número de restricciones, intentando simular condiciones de diseño más cercanas al ejercicio real.

## Agradecimientos

Los autores quieren agradecer a la Escuela de Ingeniería Civil y la Escuela de Ingeniería Mecánica de la Universidad Industrial de Santander (UIS), por la ayuda brindada para el desarrollo de esta investigación.

## Referencias

- [1] H. Assimi and A. Jamali, “A hybrid algorithm coupling genetic programming and nelder - mead for topology and size optimization of trusses with static and dynamic constraints,” *Expert Systems with Applications*, vol. 95, pp. 127–141, 2018. <https://doi.org/10.1016/j.eswa.2017.11.035> 85
- [2] O. Hasançebi, “Adaptive evolution strategies in structural optimization: Enhancing their computational performance with applications to large-scale structures,” *Computers and Structures*, vol. 86, no. 1-2, pp. 119–132, 2008. <https://doi.org/10.1016/j.compstruc.2007.05.012> 85, 91, 96
- [3] O. Hasançebi and S. K. Azad, “Adaptive dimensional search: A new metaheuristic algorithm for discrete truss sizing optimization,” *Computers and Structures*, vol. 154, pp. 1–16, 2015. <https://doi.org/10.1016/j.compstruc.2015.03.014> 85
- [4] S. Degertekin, L. Lamberti, and I. Ugur, “Discrete sizing/layout/topology optimization of truss structures with an advanced jaya algorithm,” *Applied Soft Computing Journal*, vol. 79, pp. 363–390, 2019. <https://doi.org/10.1016/j.asoc.2019.03.058> 85
- [5] K. Deb and S. Gulati, “Design of truss-structures for minimum weight using genetic algorithms,” *Finite Elements in Analysis and Design*, vol. 37, no. 5, pp. 447–465, may 2001. [https://doi.org/10.1016/S0168-874X\(00\)00057-3](https://doi.org/10.1016/S0168-874X(00)00057-3) 85

- [6] G. G. Tejani, V. J. Savsani, V. K. Patel, and P. V. Savsani, “Size, shape, and topology optimization of planar and space trusses using mutation-based improved metaheuristics,” *Journal of Computational Design and Engineering*, vol. 5, no. 2, pp. 198–214, 10 2017. <https://doi.org/10.1016/j.jcde.2017.10.001> 85
- [7] H. Cao, X. Qian, and Y. Zhou, “Large-scale structural optimization using metaheuristic algorithms with elitism and a filter strategy,” *Structural and Multidisciplinary Optimization*, vol. 57, no. 2, pp. 799–814, feb 2018. <https://doi.org/10.1007/s00158-017-1784-3> 85
- [8] M. Khatibinia and H. Yazdani, “Accelerated multi-gravitational search algorithm for size optimization of truss structures,” *Swarm and Evolutionary Computation*, vol. 38, no. June 2017, pp. 109–119, 2018. <https://doi.org/10.1016/j.swevo.2017.07.001> 85
- [9] H. Assimi, A. Jamali, and N. Nariman-zadeh, “Sizing and topology optimization of truss structures using genetic programming,” *Swarm and Evolutionary Computation*, vol. 37, pp. 90–103, dec 2017. <https://doi.org/10.1016/j.swevo.2017.05.009> 85
- [10] M. S. Gonçalves, R. H. Lopez, and L. F. F. Miguel, “Search group algorithm: A new metaheuristic method for the optimization of truss structures,” *Computers and Structures*, vol. 153, pp. 165–184, 2015. <https://doi.org/10.1016/j.compstruc.2015.03.003> 85
- [11] L. F. F. Miguel, R. H. Lopez, and L. F. F. Miguel, “Multimodal size, shape, and topology optimisation of truss structures using the firefly algorithm,” *Advances in Engineering Software*, vol. 56, pp. 23–37, 2013. <https://doi.org/10.1016/j.advengsoft.2012.11.006> 85
- [12] A. Kaveh and M. Ilchi Ghazaan, *Meta-heuristic Algorithms for Optimal Design of Real-Size Structures*, 1st ed. Cham: Springer International Publishing, 2018. <https://doi.org/10.1007/978-3-319-78780-0> 85
- [13] I. Couceiro, J. París, S. Martínez, I. Colominas, F. Navarrina, and M. Casteleiro, “Structural optimization of lattice steel transmission towers,” *Engineering Structures*, vol. 117, pp. 274–286, 2016. <https://doi.org/10.1016/j.engstruct.2016.03.005> 85
- [14] C. Tort, S. Şahin, and O. Hasançebi, “Optimum design of steel lattice transmission line towers using simulated annealing and pls-tower,” *Computers & Structures*, vol. 179, pp. 75–94, 2017. 85
- [15] R. R. de Souza, L. F. F. Miguel, R. H. Lopez, L. F. F. Miguel, and A. J. Torii, “A procedure for the size, shape and topology optimization of

- transmission line tower structures,” *Engineering Structures*, vol. 111, pp. 162–184, 2016. <https://doi.org/10.1016/j.engstruct.2015.12.005> 85
- [16] S. Degertekin, L. Lamberti, and I. Ugur, “Sizing, layout and topology design optimization of truss structures using the Jaya algorithm,” *Applied Soft Computing Journal*, 2017. <https://doi.org/10.1016/j.asoc.2017.10.001> 85, 91, 96
- [17] S. Gholizadeh and A. Baghchevan, “Multi-objective seismic design optimization of steel frames by a chaotic meta-heuristic algorithm,” *Engineering with Computers*, vol. 33, no. 4, pp. 1045–1060, oct 2017. <https://doi.org/10.1007/s00366-017-0515-0> 85
- [18] V. Mokarram and M. R. Banan, “A new PSO-based algorithm for multi-objective optimization with continuous and discrete design variables,” *Structural and Multidisciplinary Optimization*, vol. 57, no. 2, pp. 509–533, 2018. <https://doi.org/10.1007/s00158-017-1764-7> 85
- [19] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, apr 2002. <https://doi.org/10.1109/4235.996017> 86, 88
- [20] C. C. Coello and M. S. Lechuga, “Mopso: A proposal for multiple objective particle swarm optimization,” in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC’02 (Cat. No. 02TH8600)*, vol. 2. IEEE, 2002, pp. 1051–1056. 86, 89
- [21] C. A. Coello Coello, G. Toscano Pulido, and M. Salazar Lechuga, “Handling multiple objectives with particle swarm optimization,” *IEEE Transactions on evolutionary computation*, vol. 8, no. 3, pp. 256–279, 2004. <https://doi.org/10.1109/TEVC.2004.826067> 86, 89
- [22] S. Bandyopadhyay, S. Saha, U. Maulik, and K. Deb, “A simulated annealing-based multiobjective optimization algorithm: Amosa,” *IEEE transactions on evolutionary computation*, vol. 12, no. 3, pp. 269–283, 2008. <https://doi.org/10.1109/TEVC.2007.900837> 86, 89
- [23] A. Kaveh and S. Talatahari, “A particle swarm ant colony optimization for truss structures with discrete variables,” *Journal of Constructional Steel Research*, vol. 65, no. 8-9, pp. 1558–1568, 2009. <https://doi.org/10.1016/j.jcsr.2009.04.021> 86, 91, 95
- [24] A. Kaveh and M. Khayatazad, “Ray optimization for size and shape optimization of truss structures,” *Computers and Structures*, vol. 117, pp. 82–94, 2013. <https://doi.org/10.1016/j.compstruc.2012.12.010> 86



- [25] A. Kaveh and A. Zolghadr, “Comparison of nine meta-heuristic algorithms for optimal design of truss structures with frequency constraints,” *Advances in Engineering Software*, vol. 76, pp. 9–30, 2014. <https://doi.org/10.1016/j.advengsoft.2014.05.012> 86, 91
- [26] A. Kaveh and V. R. Mahdavi, “Colliding Bodies Optimization method for optimum discrete design of truss structures,” *Computers and Structures*, vol. 139, pp. 43–53, 2014. <http://dx.doi.org/10.1016/j.compstruc.2014.04.006> 86, 95
- [27] A. Mortazavi and V. Togan, “Simultaneous size, shape, and topology optimization of truss structures using integrated particle swarm optimizer,” *Structural and Multidisciplinary Optimization*, vol. 54, no. 4, pp. 715–736, 2016. <https://doi.org/10.1007/s00158-016-1449-7> 86
- [28] —, “Sizing and layout design of truss structures under dynamic and static constraints with an integrated particle swarm optimization algorithm,” *Applied Soft Computing Journal*, vol. 51, pp. 239–252, 2017. <https://doi.org/10.1016/j.asoc.2016.11.032> 86
- [29] G. Bekdas, S. M. Nigdeli, and X. S. Yang, “Sizing optimization of truss structures using flower pollination algorithm,” *Applied Soft Computing Journal*, vol. 37, pp. 322–331, 2015. <https://doi.org/10.1016/j.asoc.2015.08.037> 86
- [30] E.-G. Talbi, *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009, vol. 74. 87
- [31] P. Ngatchou, A. Zarei, and A. El-Sharkawi, “Pareto multi objective optimization,” in *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*. IEEE, 2005, pp. 84–91. 87, 88
- [32] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001, vol. 16. 87, 90
- [33] J. S. Arora, “Optimum Design Problem Formulation,” in *Introduction to Optimum Design*, 4th ed. Elsevier, 2017, pp. 19–70. 88
- [34] K. L. Du, M. Swamy *et al.*, “Search and optimization by metaheuristics,” *Techniques and Algorithms Inspired by Nature; Birkhauser: Basel, Switzerland*, 2016. 88
- [35] MATLAB, “version 9.6.0 (2019a),” Natick, Massachusetts, 2019. 90

- [36] M. Y. Cheng, D. Prayogo, Y. W. Wu, and M. M. Lukito, “A Hybrid Harmony Search algorithm for discrete sizing optimization of truss structure,” *Automation in Construction*, vol. 69, pp. 21–33, 2016. <https://doi.org/10.1016/j.autcon.2016.05.023> 91, 94
- [37] S. Fallahian, D. Hamidian, and S. M. Seyedpoor, “Optimal Design of Structures using the simultaneous perturbation stochastic approximation algorithm,” *International Journal of Computational Methods*, vol. 6, no. 2, pp. 229–245, 2009. <https://doi.org/10.1002/0471722138.ch7> 93
- [38] K. S. Lee and Z. W. Geem, “A new structural optimization method based on the harmony search algorithm,” *Computers & structures*, vol. 82, no. 9-10, pp. 781–798, 2004. <https://doi.org/10.1016/j.compstruc.2004.01.002> 93, 94