**OPTIMIZACIÓN DE ESTRUCTURAS**

**Título**

**Optimización de armaduras planas usando técnicas heurísticas**

***Title***

***Plane Truss Optimization using heuristic techniques***

**MSc. Ing. Alejandro Hernández Hernández1**

**Ing. Juan Carlos Verdecia Somoano2**

1-Universidad de Matanzas, Cuba, [alealejandro.hdezhdez@gmail.com](mailto:alealejandro.hdezhdez@gmail.com)

2-Universidad de Matanzas, Cuba, juan.verdecia@umcc.cu

**Resumen:** Actualmente las técnicas heurísticas de optimización son una opción muy utilizada como herramienta para la solución de problemas complejos de búsqueda y optimización, debido a su probada capacidad de encontrar soluciones adecuadas en un tiempo computacionalmente aceptable ya que resultan ser de las pocas técnicas efectivas en problemas que antes se resolvían gracias a la intuición y experiencia de los investigadores. Empleando estas técnicas se pueden optimizar recursos materiales, tiempo de ejecución, factibilidad, costos, entre otros, lo que facilita mejoras no solo en la calidad sino también en la economía. Una de las ramas en la que más se emplean estas técnicas es la optimización estructural, en la presente investigación se realiza una comparación del desempeño de varias de estas técnicas, así como un análisis de la mejor variante en la optimización de estructuras (tipologías de armaduras planas).

***Abstract:*** *Currently heuristic optimization techniques are a widely used option as a tool for the solution of complex search and optimization problems, due to their proven ability to find suitable solutions in a computationally acceptable time since they are among the few effective techniques in problems that before they were solved thanks to the intuition and experience of the researchers. Employing these techniques can optimize material resources, execution time, feasibility, costs, among others, which facilitates improvements not only in quality but also in the economy. One of the branches in which these techniques are most used is the structural optimization, in the present investigation a comparison of the performance of several of these techniques is made, as well as an analysis of the best variant in the optimization of structures (types of reinforcements flat.*

**Palabras Clave:** Optimización de armaduras, técnicas heurísticas de optimización.

***Keywords:*** *Truss optimization; heuristics optimization methods*.

**1. Introducción**

Hasta las últimas décadas del pasado siglo, muchos de los procesos de diseño fueron realizados por la experiencia e intuición del diseñador, en muchas ocasiones derivadas de la subjetividad, en vez de una aplicación intensiva de la teoría de optimización o la simulación. Recientemente esta forma de pensar ha cambiado debido a la importancia que ha tomado el campo de la optimización estructural en el diseño estructural, ya que mediante su aplicación se logra reducir costos, materiales y tiempo en los procesos de diseño realizados por los equipos de proyecto. El propósito de aplicar los conceptos de diseño óptimo a la ingeniería estructural es el de obtener una solución a un problema que cumpla con todas las limitaciones y restricciones impuestas, y que a la vez resulte ser la mejor en cuanto a uno o varios criterios de diseño previamente establecidos ([Chen, 2013](#_ENREF_11)).

Una de las maneras de enfocar la optimización estructural es a partir de la programación genética, en la inteligencia artificial, el término programación genética se refiere a una metodología basada en los algoritmos evolutivos e inspirada en la evolución biológica para desarrollar automáticamente programas de computadoras que realicen una tarea definida por el usuario. Es una técnica de aprendizaje automático utilizada para optimizar un grupo de individuos (por ejemplo tipologías de armaduras o celosías) evaluando su capacidad de acuerdo a una función de ajuste o aptitud, permitiendo al usuario elegir la solución más idónea. Como [metodología científica](zim://A/A/html/M/e/t/o/Metodolog%C3%ADa_cient%C3%ADfica.html), la heurística es aplicable a cualquier [ciencia](zim://A/A/html/C/i/e/n/Ciencia.html) e incluye la elaboración de medios auxiliares, principios, reglas, estrategias y programas que faciliten la búsqueda de vías de solución a problemas; o sea, para resolver tareas de cualquier tipo para las que no se cuente con un [procedimiento algorítmico](zim://A/A/html/A/l/g/o/Algoritmo.html) de solución. La importancia de minimizar el número de soluciones posibles está enfatizada en generar problemas razonables de optimización que se refieren generalmente en la optimización estructural a las tres categorías del diseño: el tamaño, la forma y la topología.

Esta disciplina ha ido evolucionando con el paso del tiempo hasta convertirse en un área de ingeniería llamada optimización estructural. El interés creciente en esta área en las últimas décadas es debido a la disponibilidad de computadoras baratas y poderosas, junto con los desarrollos rápidos en los métodos de análisis estructural y de optimización que juega un papel principal en muchos campos de ingeniería. Por otra parte numerosos problemas a los que se enfrentan los ingenieros civiles están descritos por la optimización de estructuras.

La optimización de estructuras (armaduras, celosías, entre otras) se ocupa de la selección de nodos y su conectividad, en otro nivel, crea búsquedas de optimización para encontrar las coordenadas óptimas de nodos existentes, finalmente, las secciones transversales de miembros de la estructura pueden ser optimizadas. En cada caso, el problema de optimización es subordinado a algunas restricciones en los desplazamientos del nodo, las cargas críticas de alabeo, entre otras. La función objetivo es en muchos casos minimizar el peso de estructura que se correlaciona con el costo global.

Elegir la mejor solución y mejorarla, depende fuertemente de la experiencia del diseñador, por lo que no es simple y no se dispone de los parámetros o la capacidad suficientes para asegurar que la última propuesta de solución que se obtenga sea la óptima, ya que es físicamente imposible evaluar el espectro completo de posibles soluciones. Entonces, resulta atractivo tener una herramienta que permita obtener soluciones de un problema de diseño lo más cercanas posibles a la óptima, cuyo desempeño no dependa de la experiencia del diseñador, de lo que se deriva el **problema** de que los diseños de las diferentes tipologías de armaduras planas que se realizan hoy en Cuba, se obtienen fundamentalmente basados en experiencias y criterios del diseñador y no de un proceso de optimización que derive en armaduras más racionales y económicas, lo que implica un gasto innecesario de recursos.

El concepto de **optimización**, que se refiere a la selección del mejor elemento de un conjunto de elementos posibles, de acuerdo a un criterio de selección dado. Para esto, se deben emplear métodos de optimización que permitan conocer los mejores valores que puede asumir dicho conjunto de variables.([Gong et al., 2014](#_ENREF_20)) ([Coello and Christiansen, 2000](#_ENREF_7)) ([Coello et al., 1995](#_ENREF_5))

La **función objetivo**, es la función que mide cualitativamente el funcionamiento del sistema en un proceso de optimización, de modo que lo que se busca es la maximización o minimización de esta, enmarcado en una gama de posibles soluciones.

Una **restricción** es un parámetro que permite ceñir, circunscribir o reducir a un límite un elemento determinado, de modo que haya validez en el diseño.([Coello et al., 2002](#_ENREF_8)). Dado que estas restricciones son aplicadas en problemas de tipo físico y matemático, se pueden clasificar en dos grandes grupos, funcionales y geométricas, las primeras se refieren al comportamiento físico-mecánico del sistema, las otras a las dimensiones y tipología de los elementos.

Se define como **espacio de búsqueda** al conjunto de todos los elementos que pueden ser operados por los métodos de optimización. La búsqueda generalmente se realiza con los mejores individuos de distintas áreas del espacio del problema. ([MacDonald, 2005](#_ENREF_30)).

Las **variables** representan las decisiones que se pueden tomar para variar el valor de la función objetivo. Se pueden clasiﬁcar como dependientes o independientes. En el caso de la optimización estructural de armaduras planas o celosías por lo general son consideradas como variables las longitudes y secciones transversales de los elementos, es decir la posición de los nudos y la rigidez de las barras.([Pires, 2014](#_ENREF_33)), ([Coello et al., 2002](#_ENREF_8)), ([Coello, 2012](#_ENREF_6), [Deshpande et al., 2013](#_ENREF_16)).

Los **Algoritmos Evolutivos** incluyen técnicas relacionadas con la optimización metaheurística. Con el auge que tomaron estas teorías, y por medio de los descubrimientos de nuevas tecnologías que permitían un desarrollo computacional y matemático mucho más elevado, empezaron a investigarse teorías y algoritmos artificiales sentando sus bases en los mecanismos naturales con los trabajos de Holland. ([Holland, 1975](#_ENREF_23)).

A partir de esta fecha, los estudios sobre algoritmos evolutivos, en cualquiera de sus facetas, se han incrementado de tal forma que ya es difícil encontrar un área de investigación en la que no hayan irrumpido con fuerza. Se han encontrado implementados en ámbitos tan diversos como finanzas, inversión, educación, transporte, redes de comunicación, planificación, logística, producción, optimización, entre otras.([Biethan and Nissen, 1995](#_ENREF_4)). Las primeras ideas en las que se basan técnicas heurísticas se pueden encontrar en los artículos de Holland de principios de los años 1960. En ellos se establecen los aspectos básicos para la comprensión de los principios de los sistemas adaptativos. Estos sistemas son capaces de automodificarse en respuesta a su interacción con el medio en el que están funcionando. Las teorías sobre los sistemas adaptativos están orientadas a facilitar, por un lado, la comprensión de las formas complejas de adaptación que aparecen en los sistemas naturales y, por otro, la habilidad para diseñar sistemas adaptativos robustos. A mediados de los años 1960, las ideas de Holland empezaron a plasmarse en modelos implementados en ordenador. En cada uno de estos sistemas se representaban elementos por medio del uso de genomas en los que los mecanismos de evolución y herencia eran abstracciones de operadores genéticos como la mutación, el cruce o la inversión ([Holland, 1975](#_ENREF_23)), ([Saka and Greem, 2012](#_ENREF_34)), ([Goldberg, 1989](#_ENREF_19)).

A partir de la década de 1980 fueron aumentando la cantidad y complejidad de los problemas de optimización de estructuras que son abordados a partir del empleo de las heurísticas de optimización, un gran número de estos problemas son de armaduras planas y espaciales.([Cheng, 2010](#_ENREF_13)), ([Ahrari et al., 2014](#_ENREF_1)), ([Gutierrez et al., 2012](#_ENREF_21)), ([Hultman, 2010](#_ENREF_25)), ([Smith et al., 2002](#_ENREF_35)).

**Población:** Este parámetro indica el número de cromosomas que se tiene para una generación determinada. Es importante que exista una población lo suficientemente amplia, para que haya un adecuado número de reproducciones y se llegue a una solución óptima. No obstante, el tamaño de la población debe definirse de modo que no sea demasiado amplio y por lo tanto la convergencia tome mucho tiempo. Por lo general, la población inicial se genera como una combinación aleatoria de individuos a partir de un rango de variables de diseño([Chen and Zou, 2014](#_ENREF_12)).

**Función de evaluación (*Fitness*):** Esta función es la que vincula el algoritmo con el problema de optimización y es la que evalúa la calidad de la solución. Comienza tomando un cromosoma de entrada y retorna un número o conjunto de números que dan una medida de la calidad del cromosoma en relación a las condiciones del problema, ([Coello et al., 2002](#_ENREF_8)).

**Selección**: Es el operador que selecciona un grupo de individuos de la población actual para el apareamiento, dependiendo del valor de su *fitness.* ([Saka and Greem, 2012](#_ENREF_34))

**Cruzamiento o Reproducción**: Esta es la segunda operación genética, en la cual los *strings* de los padres seleccionados son fraccionados y algunos genes son intercambiados para generar nuevos individuos. ([Kona et al., 2006](#_ENREF_27))

**Mutación**: En la evolución, una mutación es un suceso bastante poco común. Estas contribuyen a la diversidad genética de la especie. En un algoritmo evolutivo tendrán el mismo papel, y la misma frecuencia es decir, muy baja. Una vez aplicado el cruzamiento para obtener una nueva población se aplica este tercer operador genético, la mutación, que consiste en alterar aleatoriamente los genes de algunos de los individuos de la población. Si no hay mutación, los descendientes son los mismos que había cuando se dio la reproducción. En caso de que haya mutaciones, parte del cromosoma descendiente es modificado y si la probabilidad de mutación es del 100%, significa que la totalidad del cromosoma es alterada. ([Koziel and Yang, 2011](#_ENREF_28))

**Elitismo**: Este operador se emplea para que las operaciones de cruce y mutación no afecten un buen cromosoma indefinidamente. Lo que hace es copiar el mejor individuo de la población anterior a la siguiente, lo que permite una mejor convergencia del algoritmo. Sin embargo, deben imponerse las restricciones necesarias para que no se generen “superindividuos”. ([Marler and Arora, 2004](#_ENREF_31)), ([Allaire et al., 2004](#_ENREF_2)).

Los algoritmos heurísticos por lo general tienen un tiempo de ejecución mucho menor y evidencian la posibilidad de mejorar significativamente el diseño de las estructuras, a partir de una línea de desarrollo que se impone en el mundo, ([Vidal, 2013](#_ENREF_37)).

Dentro del grupo de los algoritmos heurísticos se pueden distinguir los denominados métodos metaheurísticos, que imitan fenómenos simples observados en la naturaleza y que están asociados con la inteligencia artiﬁcial. Estos algoritmos (evolutivos) tratan de adaptar el comportamiento de diferentes especies a soluciones de problemas altamente complejos mediante optimización.([Chen, 2013](#_ENREF_11)), ([Coello et al., 2002](#_ENREF_8)), ([Vamplew et al., 2010](#_ENREF_36), [Hultman, 2010](#_ENREF_25)), ([Hultman, 2010](#_ENREF_25)), ([Coello et al., 2002](#_ENREF_8))

Uno de los elementos estructurales más usados en la actualidad son las **cerchas o armaduras**, las cuales soportan cargas elevadas y cubren grandes luces, generalmente se utilizan en cubiertas de techos y puentes.

Es una estructura reticulada simple formada por elementos rectos de sección constante, cuya longitud supera varias veces su sección transversal, se conocen como barras y se conectan rígidamente en sus extremos denominados nodos o nudos, los esfuerzos actúan a lo largo de su eje longitudinal. Las armaduras planas o cerchas se utilizan para soportar cargas elevadas y cubrir grandes luces, pueden construirse en maderas o acero y usadas en cubiertas de techos, puentes, grúas, torres, entre otros.

El ***software*** MATLAB, de *Mathwork*, se expone como una herramienta fuerte dentro del campo de la optimización y el cálculo matemático. Las funciones de Matlab se agrupan en más de 35 cajas de herramientas y paquetes de bloques, dentro de los que se encuentra el de Matemáticas y Optimización. Por sus características es un software utilizado ampliamente en el mundo académico y científico, dentro del cual se destaca el campo de la optimización mediante algoritmos evolutivos([Coello, 2012](#_ENREF_6)), ([Coello et al., 2002](#_ENREF_8)).

Po otra parte el programa SAP 2000 es uno del software líder en la ingeniería estructural: se pueden analizar cualquier tipo de estructuras con este programa e incluso diseñar elemento por elemento de manera precisa con los reglamentos más conocidos (ACI, EUROCODIGO y otros). Se trata de un excelente programa de cálculo estructural en tres dimensiones mediante elementos finitos. Es el descendiente directo de la familia SAP90, muy conocida hace algunos años. En este caso, el programa está totalmente renovado. Tal vez lo más visible sea su nueva interfaz, totalmente integrada en *Windows* y realmente sencilla de usar. Mediante SAP2000 es posible modelar complejas geometrías, definir diversos estados de carga, generar pesos propios automáticamente, asignar secciones, materiales, así como realizar cálculos estructurales de hormigón y acero basados en variadas normativas.([Chavez, 2018](#_ENREF_10)).

**Técnicas heurísticas empleadas para el estudio**

***Not sorting genetic algorithms* II (NSGAII):** Los algoritmos genéticos son métodos metaheurísticos de optimización basados en un planteamiento probabilístico que además realizan la búsqueda por poblaciones. Su importancia radica en que es un método capaz de converger con muy alta probabilidad en un óptimo global, a diferencia de los métodos basados en gradientes que eran los que más se utilizaban hasta la aparición de estos nuevos algoritmos. Algunos autores han reportado que NSGA (*Not sorting genetic algorithms*) tienen un rendimiento algo más bajo que MOGA (*multi-objective genetic algorithms*), además de que NSGA se acerca más a la frontera de Pareto que sus predecesores en algunos tipos de problemas. El NSGA-II parece ser notablemente más eficiente que su predecesor NSGA, pero también presenta una cuestionable capacidad exploratoria, además el algoritmo tiende a esparcirse rápidamente y de modo apropiado en ciertas regiones con soluciones no dominadas sin embargo parece presentar problemas en regiones con características distintas a estas dentro del espacio de búsqueda, también se ha demostrado que no es muy efectivo a medida que aumenta el número de objetivos.

***Not sorting genetic algorithms* III (NSGAIII)** Las funciones con muchos objetivos se presentan como un reto para los algoritmos evolutivos basados en la dominancia de Pareto clásica (MOEAs *Multi objective evolutionary algoritms*) incluido NSGA II, esto es debido a que la calidad de los resultados, es decir la posibilidad de los algoritmos de acercarse o predecir la frontera real de Pareto se hace más baja con el aumento de la cantidad de objetivos a analizar. Más recientemente ha surgido NSGA III este parece tratar mejor los problemas multiobjetivos, en este caso se mantiene la diversidad entre los miembros de la población mediante el suministro y la actualización adaptativa de una serie de puntos de referencia bien distribuidos, sin embargo todavía depende de la dominancia de Pareto para acercar la población hacia el frente de Pareto, dejando así la posibilidad de la ocurrencia de la no convergencia.

***Multi Objective Cross Entropy* (MOCE+):** Las ventajas de los algoritmos de estimación de distribución (Estimation-of-Distribution algorithms) y los métodos específicos de entropía cruzada (*crossentropy*) son ampliamente reconocidos([Giagkiozis et al., 2014](#_ENREF_18)). (MOCE+) es una propuesta de este tipo de algoritmos, el cual está basado en un nuevo procedimiento para abordar las restricciones: (i) el uso de valores límites para seleccionar la población elitista, filtrando esta población después de cada iteración.([Beruvides et al., 2016](#_ENREF_3))

***Simple Multi Objective Cross Entropy* (SMOCE):** Este algoritmo *Simple Multi-Objective Cross-Entropy method* (SMOCE) está basado en su antecesor MOCE+, como en todas las técnicas de optimización multiobjetivo SMOCE se centra en determinar los mínimos de cualquier función, el algoritmo base de este método se representa a continuación([Haber et al., 2017](#_ENREF_22)). El proceso evolutivo en este algoritmo tiene lugar en un solo lazo con una única condición de parada (basado en *epoch number*, N), estas son dos de las ventajas de SMOCE sobre su antecesor MOCE+ donde eran considerados dos lazos y tres condiciones de parada, esta modificación permite la reducción de la cantidad de evaluaciones de la función objetivo y la eliminación de los parámetros de máximo número de evaluaciones y el límite de convergencia. En este algoritmo en la primera iteración se trabaja con una población creada al azar y en las siguientes iteraciones se va creando una población a partir de la anterior y se evalúa los valores de la función objetivo después de creada la población. En este caso las mejores soluciones (*elite solutions*) son incluidas en la próxima población lo cual introduce elitismo en el algoritmo (SMOCE), presentándose esto como otra ventaja de este método sobre MOCE+([Haber et al., 2017](#_ENREF_22)).

***Multi-objective evolutionary algorithm based on decomposition*(MOEAD):** Normalmente las diferentes variantes de MOEA/D (*multi-objective evolutionary algorithm based on decomposition*) no presentan un mecanismo de normalización, debido a esto muestran dificultades para ajustarse correctamente al frente real de Pareto de los problemas tratados, es por ello que algunas de las variantes más recientes de MOEA/D si presentan este mecanismo de normalización([Ke et al., 2013](#_ENREF_26)). Este algoritmo está basado en descomponer en varios problemas de un solo objetivo el problema real que es de hecho un problema multiobjetivo. El objetivo de cada subproblema es una agregación ponderada (lineal o no lineal) de los objetivos individuales, la relación de cada uno de los problemas con sus vecinos son definidas a partir de sus vectores de ponderación, de este modo cada subproblema es optimizado a partir de la información de sus vecinos ([Ke et al., 2013](#_ENREF_26)).

***Multi objective particle swarm optimization* (MOPSO):** *Particle swarm optimization***(**PSO) es un método computacional que intenta optimizar un problema a través de iteraciones sucesivas, tratando de obtener los mejores candidatos de solución, a partir de cierta medida de calidad.([Derrac et al., 2011](#_ENREF_15)) En esta técnica las posibles soluciones o partículas se van moviendo por el espacio de búsqueda a medida que sus posiciones son evaluadas, de modo que las partículas que alcanzan mejores posiciones (mejores soluciones) atraen al resto hacia esos espacios, logrando así encontrar las mejores ubicaciones posibles, lo más cercano a la frontera real de Pareto como puedan([Coello and Zalazar, 2003](#_ENREF_9), [Pedersen and Chipperfield, 2010](#_ENREF_32)).

***Strength Pareto Evolutionary Algorithm*** (**SPEA-II**) El SPEA fue introducido por Eckart Zitzler y Lothar Thiele, esta técnica fue concebida como un modo de integrar diferentes algoritmos evolutivos. SPEA usa un archivo externo conteniendo soluciones no dominadas previamente encontradas. En cada generación, los individuos no dominados son enviados para el conjunto no dominado inicial (externo). Para cada individuo en este set externo, un valor de desempeño es computado. El *fitness* de cada miembro de la población actual es computado según el desempeño de todas las soluciones no dominadas externas que lo dominan([Zitzler et al., 2001](#_ENREF_38)). En este algoritmo se usa el criterio de la dominancia de Pareto para asegurar que la población está correctamente esparcida a lo largo de la frontera de Pareto. *(*[*Coello et al., 2002*](#_ENREF_8)*).* SPEA2 tiene tres diferencias fundamentales con respecto a su predecesor SPEA. Incorpora una refinada técnica de *fitness* o adaptación que tomo en cuenta para cada individuo el número de individuos que lo dominan a él y el número de individuos que el domina. Usa como técnica la llamada *a nearest neighbor density estimation* (estimación de la densidad de vecino más cercano) lo cual conduce la búsqueda de modo más eficiente.Tiene un método mejorado de truncamiento que garantiza la preservación de soluciones cercanas a la frontera. ([Coello et al., 2002](#_ENREF_8)).

***Pareto Envelope-based Selection Algorithm* (PESA II):** El *Pareto Envelope-based Selection Algorithm* (PESA) fue desarrollado por Corne. Este algoritmo (PESA) está compuesto por una población externa grande y una población interna considerablemente menor. Se utiliza una división tipo rejilla (muy fina) del espacio para mantener la diversidad de la selección mientras se ejecuta el MOEA, esta medida se utiliza para permitir soluciones en la población externa a través de un archivo de soluciones que evalúan vectores no dominados ([Coello et al., 2002](#_ENREF_8)). En PESA II se incluye una diferencia con respecto a PESA, en este caso la selección se hace por regiones y los sujetos son seleccionados no individualmente, sino por paquetes o *hiperbox*, este cambio está motivado por el propósito de disminuir el costo computacional asociado al *ranking* de Pareto ([Li et al., 2013](#_ENREF_29)).

**Funciones de prueba:** Este término se refiere a funciones preestablecidas o generadas a partir de ciertas condiciones, las mismas son muy variadas y van desde muy sencillas hasta muy complejas, su principal característica y razón de su existencia es que se conoce su frontera real de Pareto, lo que las hace muy útil a la hora de evaluar el desempeño de las diferentes heurísticas, pues se puede determinar con qué nivel de precisión son capaces de encontrar este frente de Pareto cada una de las técnicas de búsqueda ([Huband et al., 2006](#_ENREF_24)), ([Deb et al., 2001](#_ENREF_14)).

Tabla 1 Funciones de prueba, dominio y frente real de Pareto

|  |  |  |  |
| --- | --- | --- | --- |
| MOP1 |  |  | MOP1.png |
| MOP2 |  |  | MOP2.png |
| MOP3 |  |  | MOP3.png |
| MOP4 |  |  | MOP4.png |
| MOP6 |  |  | MOP6.png |
| ZDT1 |  |  | ZDT1.png |
| ZDT2 |  |  | ZDT2.png |
| ZDT3 |  |  | ZDT3.png |
| WFG2 |  |  | WFG2.png |
| WFG3 |  |  | WFG3.png |
| WFG4 |  |  | WFG4.png |
| WFG5 |  |  | WFG5.png |
| WFG6 |  |  | WFG6.png |

Parámetros que influyen en el desempeño de las heurísticas: Todas las simulaciones realizadas se hicieron en una computadora personal con Intel Core i3-2120 CPU (3.30 GHz) and 4.0 GB RAM. Durante el tiempo que duraba el experimento con cada algoritmo solo se dedicaba la computadora a esta tarea, no realizándose ninguna otra que pudiera afectar el tiempo de ejecución, uno de los parámetros a medir en el estudio.

En todos los casos el objetivo es obtener para cada problema de prueba, usando cada uno de los algoritmos antes mencionados el desempeño de estos últimos en cuanto a la calidad de los resultados (aproximación a la frontera real de Pareto) y tiempo de ejecución, para ello se efectuaron varias iteraciones, variando solamente los parámetros cantidad de generaciones (G o Z) y tamaño de la población (N) hasta obtener los mejores resultados posibles (soluciones no dominadas)([Chen and Zou, 2014](#_ENREF_12)).

**Métricas para realizar la comparación:** En este caso se usan como parámetros para medir el desempeño de los algoritmos ante las funciones de prueba el tiempo de ejecución o de cómputo y la relación de hiperárea, este último representa la calidad de los resultados pues es una buena medida de que tan cercano se encuentra la frontera de Pareto alcanzada por el algoritmo y la frontera real de Pareto.

En las funciones MOP en general la mayoría de las heurísticas tienen buenos desempeños solo PESA II tiene un desempeño marcadamente más bajo que el resto y en algunos casos MOEAD también tiene un desempeño algo menor que el resto, en cuanto al tiempo de ejecución (relacionado al costo computacional) son MOCE+ y su sucesor SMOCE las que tienen mejores resultados, pues además de tener un muy buen desempeño (HR) lo logran empleando muy poco tiempo.

En este tipo de funciones (ZDT) en general el desempeño de todas las heurísticas disminuye y aumenta el tiempo de ejecución, en este caso no se muestra una marcada tendencia de que alguna heurística sea mejor que otra sino que son ligeramente superiores NSGA II, MOCE+ y SMOCE sobre el resto.

En el caso de las funciones WFG se nota que SMOCE es la técnica con mejor desempeño pues alcanza los mayores valores de HR empleando valores de tiempo de ejecución similares al resto.

**Análisis del caso de optimización (configuración de armaduras planas):** Las armaduras planas, entre ellas las de tipología *Pratt* son de las estructuras más utilizadas en la actualidad por su ya probada eficacia, tanto en puentes, cubiertas, entre otras.

En todos los casos las funciones objetivos resultan las tensiones en las barras de las armaduras y la masa total de estas estructuras, debido a que disminuyendo lo más posible estos valores se obtendría secciones transversales más pequeñas lo que influye directamente en el costo relacionado con el gasto de materiales para su ejecución.

 ecuación 1

La masa total de la armadura depende directamente de la densidad del material (γ) y del volumen de las barras que la componen (*Asec* se refiere al área de la sección transversal del elemento, *ln* longitud de la barra).

En el caso de la tipología *Pratt* las variables que intervienen en la función son las alturas en los extremos (longitud total del primer montante), la altura de cada montante, la longitud de todos los elementos del cordón inferior, lo cual fija por supuesto las longitudes de los elementos del cordón superior. Obtener una configuración racional, resistente y económica ha sido una tarea difícil a la que se enfrentan los ingenieros pues muchas son las variables a tener en cuenta, siendo la experiencia y criterio del diseñador una parte importante del proceso, sin embargo, en muchas ocasiones se dejan de analizar soluciones que pueden ser sino óptimas al menos mejor que la propuesta definitiva, pues nunca se tuvo en consideración.

Sin embargo, con el uso de los algoritmos propuestos estas soluciones no escaparían a la vista de los que diseñan y se pueden ahorrar considerables recursos solo con el uso de estas técnicas, por esto es que se propone una programación en MatLab que permita a los diseñadores hacer uso de ella para ajustarla a las condiciones de cada problema y así obtener las mejores soluciones posibles. A continuación, se propone un problema tipo para validar el funcionamiento de los algoritmos propuestos, los valores de longitudes totales y cargas pueden ser cambiados según las necesidades del diseñador para este estudio se mantienen los valores en cada caso pues el objetivo es comparar los resultados obtenidos por criterio de diseñador y usando los algoritmos propuestos. En este se requiere salvar una longitud L (en este caso de 50 m), con apoyos extremos (simplemente apoyados) y sin apoyos intermedios; usando la menor variedad de perfiles posibles para una mejor facilidad de armado. Para ello se utiliza una carga distribuida q (de 25 kN/m en este caso, puede también usarse cargas puntuales en los nodos que se desee).

**Solución propuesta por el diseñador (Pratt):**

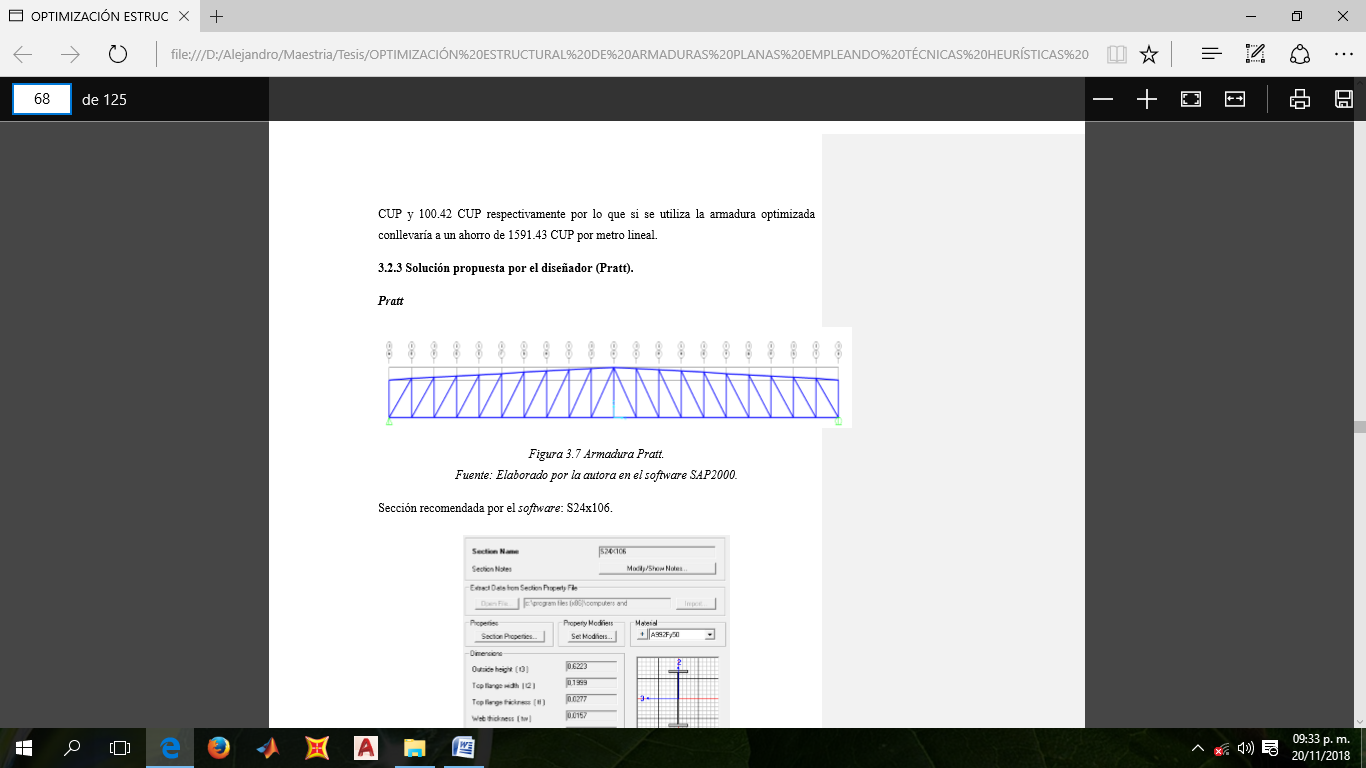


Figura 1 Armadura Pratt. Fuente: Elaborado por el autor en el software SAP2000.

Sección recomendada por el software: S24x106.

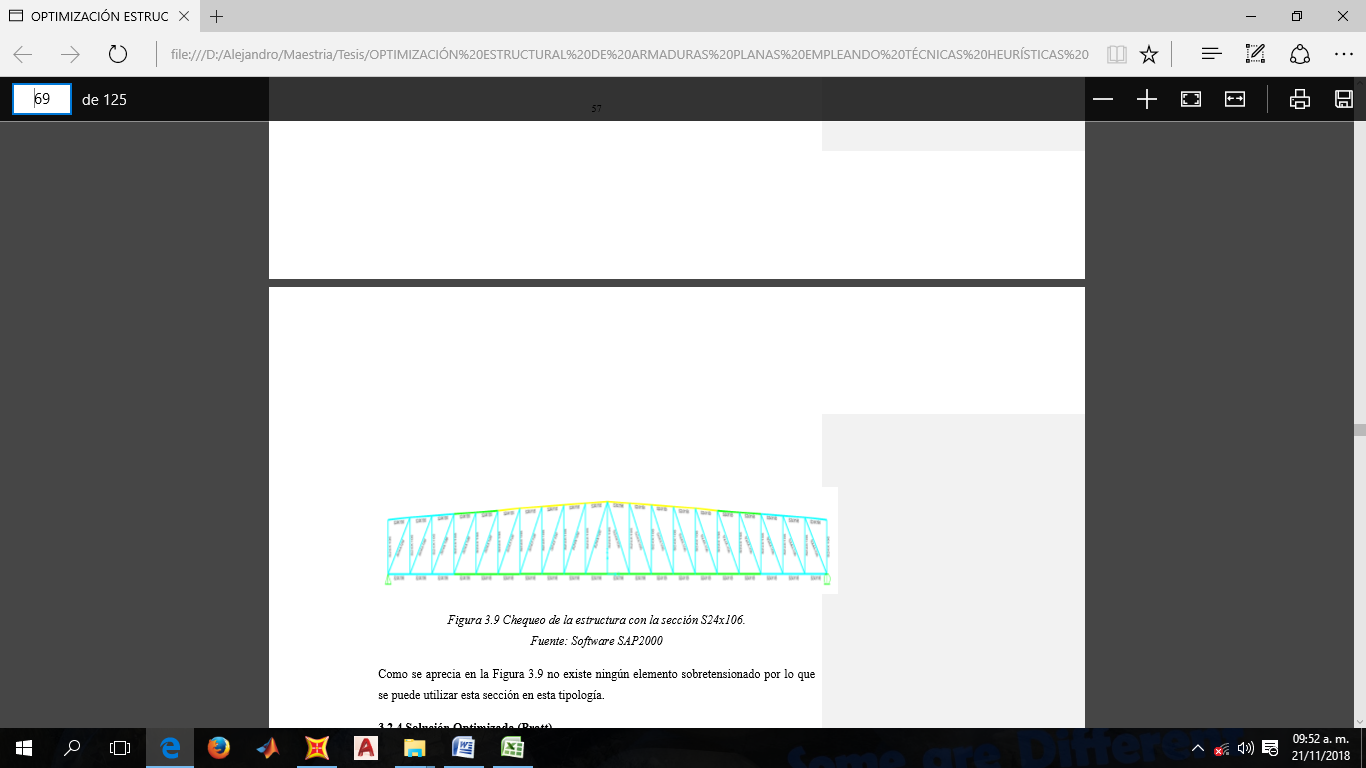


Figura 2 Chequeo de la estructura con la sección S24x106. Fuente: Software SAP2000

Como se aprecia en la Figura 2 no existe ningún elemento sobretensionado por lo que se puede utilizar esta sección en esta tipología. Conclusión a la que puede arribar el diseñador al usar este modelo estructural.

**Solución Optimizada con NSGA II (Pratt):** A continuación se muestra la frontera de Pareto que se obtiene usando NSGA II, en este caso se aprecia, al igual que en la tipología anterior, cómo se obtienen variantes en las cuales la masa es considerablemente grande y las tensiones bajas (en el extremo superior izquierdo del gráfico) y en el extremo opuesto sucede lo contrario, además se aprecia una zona del gráfico donde se agrupan soluciones con tensiones y masas medias, que corresponden a soluciones con una configuración satisfactoria

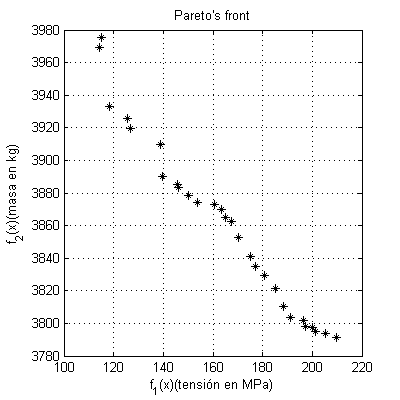


Figura 3 Frontera de Pareto de la tipología Pratt usando NSGA II

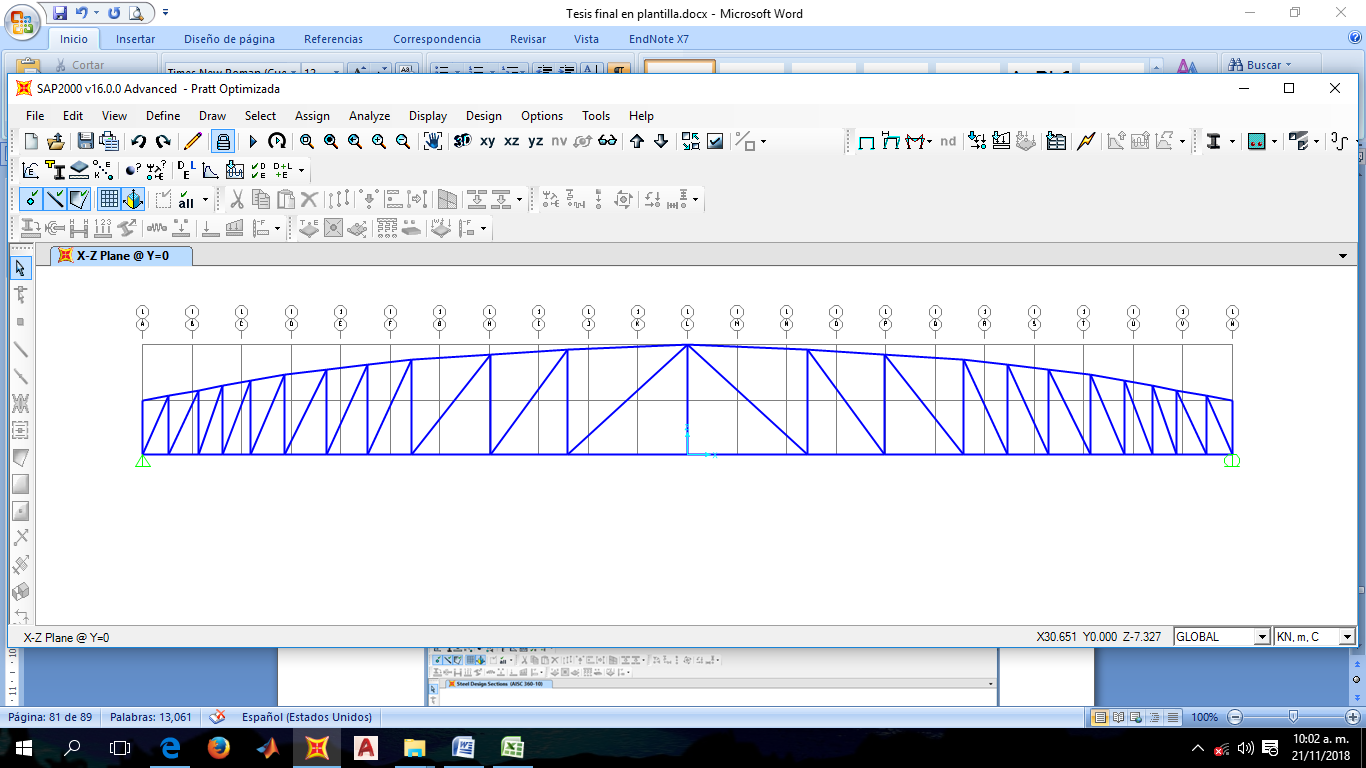


Figura 4 Armadura Pratt Optimizada. Fuente: Elaborado por el autor en el software SAP2000.

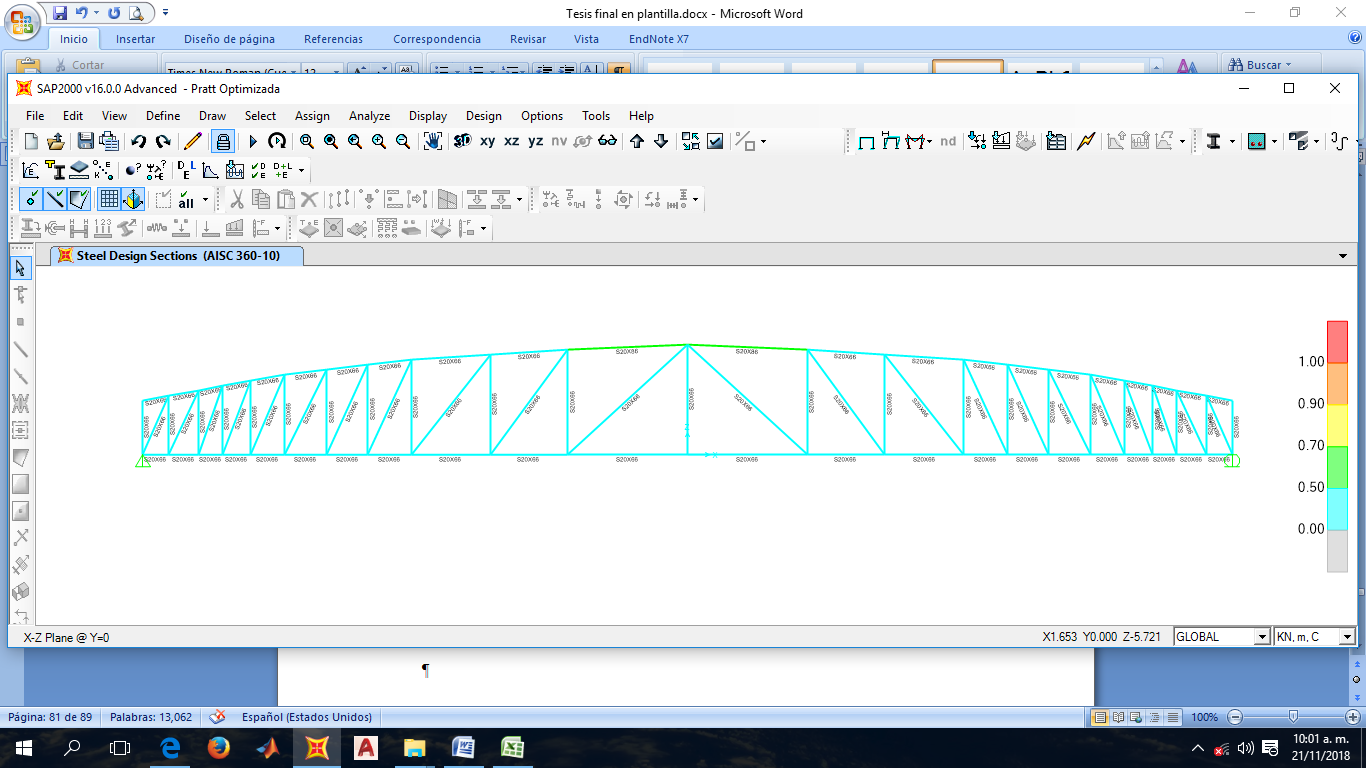


Figura 5 Chequeo de la estructura con la sección S20x66. Fuente: Software SAP2000

Como se aprecia en la Figura 3.19 no existe ningún elemento sobretensionado por lo que se puede utilizar esta sección en esta tipología.

En este caso también se logra una mayor disminución en el área de los perfiles usando NSGA II, pues por experiencia se usa una sección (S24x106), con un área de 0.0201 m2, y con NSGA II una sección S20x66 con área de sección transversal de 0.0125 m2.

**Solución Optimizada con SMOCE (Pratt):** A continuación se muestra la frontera de Pareto obtenida con SMOCE de la tipología *Pratt,* en este caso el comportamiento es similar a las fronteras obtenidas hasta el momento.

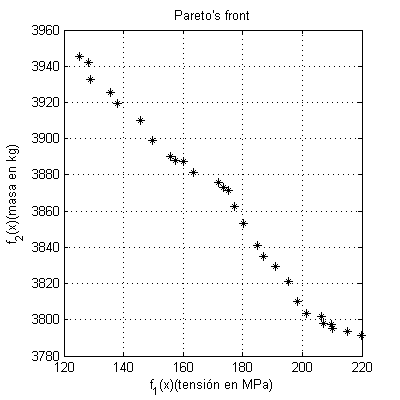


Figura 6 Frontera de Pareto de la tipología Pratt usando SMOCE

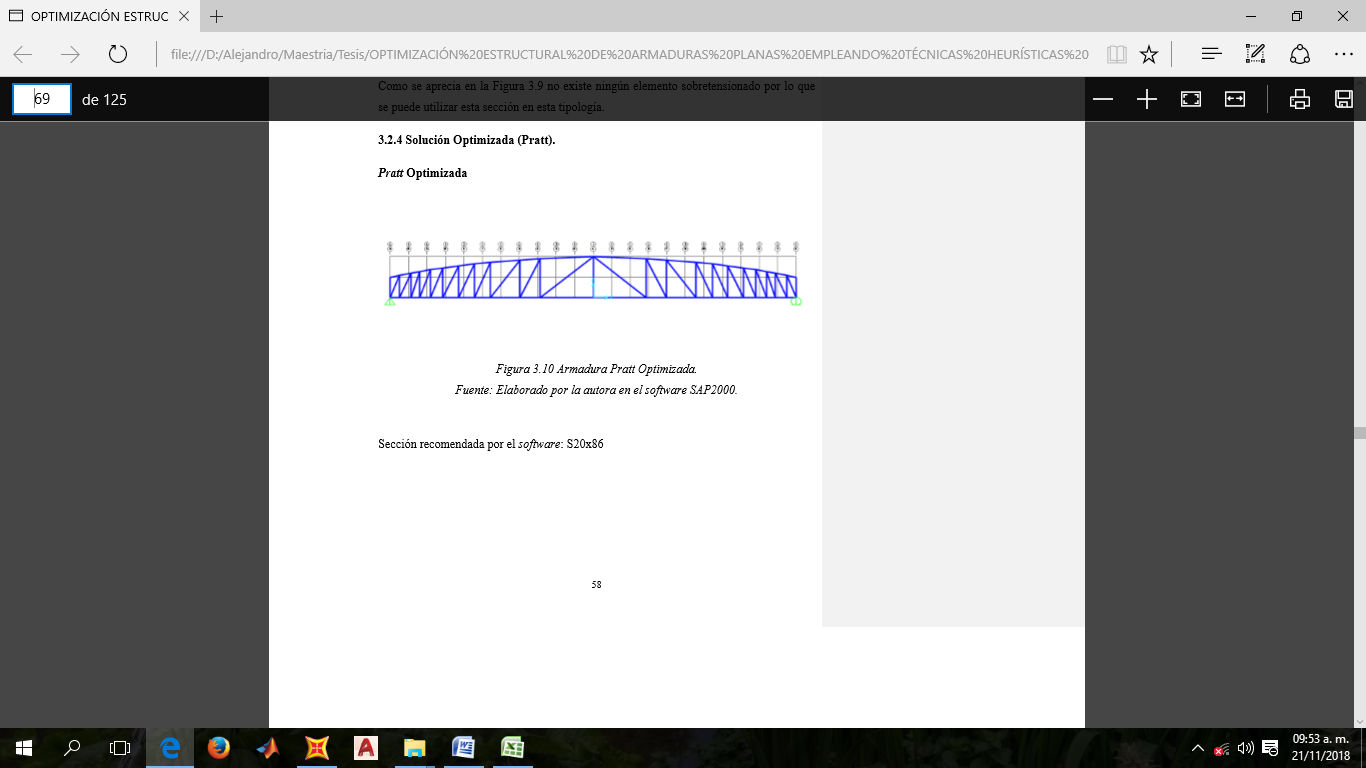


Figura 7 Armadura Pratt Optimizada. Fuente: Elaborado por el autor en el software SAP2000.

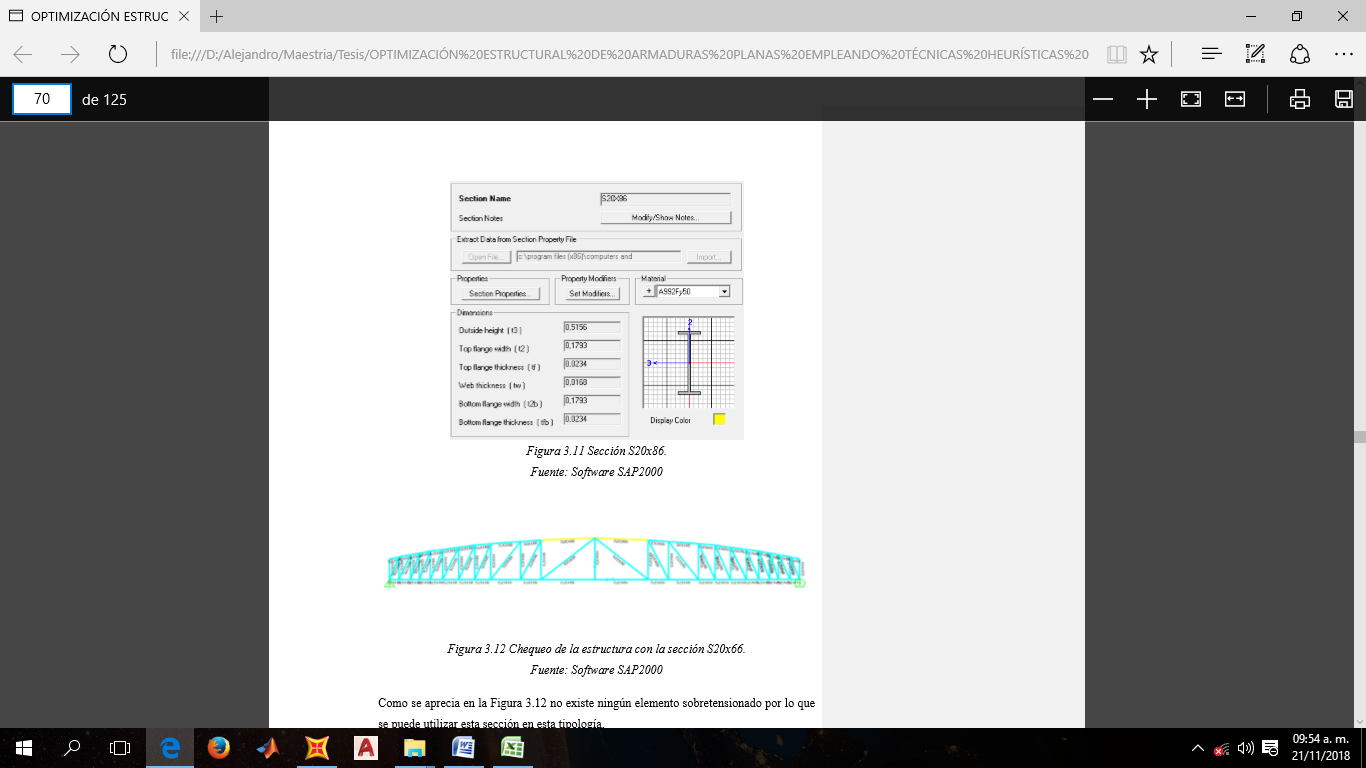


Figura 8 Chequeo de la estructura con la sección S20x86. Fuente: Software SAP2000

Como se aprecia en la Figura 3.17 no existe ningún elemento sobretensionado por lo que se puede utilizar esta sección en esta tipología.

Al igual que en la Sloped si se comparan ambas soluciones se puede apreciar que en la propuesta optimizada con NSGA II se utiliza una menor sección (S20x86), con un área de 0.0163 m2, mientras que en la propuesta por experiencia del diseñador se usa una sección mayor (S24x106), con un área de 0.0201 m2, aunque la sección obtenida con SMOCE es algo más grande que la obtenida con NSGA II.

Tabla 2 Comparación de los resultados del desempeño de las técnicas empleadas en la tipología Pratt

|  |  |  |  |
| --- | --- | --- | --- |
| Técnica | Sección | Área (m2) | Disminución % respecto a la por experiencia |
| Experiencia | S24x106 | 0.0201 | - |
| SMOCE | S20x86 | 0.0163 | 18.90547264 |
| NSGA II | S20x66 | 0.0125 | 37.81094527 |

**Conclusiones**

Las heurísticas de optimización se han convertido en la actualidad en unas de las técnicas más difundidas y utilizadas en diversos campos, siendo uno de ellos la optimización estructural, especialmente las estructuras reticuladas planas.

Los resultados de las diferentes heurísticas seleccionadas varían según la complejidad del problema, en sentido general se destacan tres de ellas como las de mejor rendimiento ante estos problemas (SMOCE, MOCE+ y NSGA III).

Es evidente la mejora en la racionalidad del diseño cuando se emplean algoritmos de búsqueda heurística a problemas de armaduras planas (*Pratt*) pues se logran secciones transversales más pequeñas que redundan finalmente en estructuras más ligeras y económicas, a la vez que son resistentes a las cargas impuestas.

**Referencias bibliográficas**

AHRARI, A., ATAI, A. & DEB, K. 2014. Simultaneous Topology, Shape and Size Optimization of Truss Structures by Fully Stressed Design Based on Evolution Strategy.

ALLAIRE, G., JOUVE, F. & TOAEDE, A. 2004. Structural optimization using sensitivity analysis and a level-set method. . *Computational Physics.*

BERUVIDES, G., QUIZA, R. & HABER, R. 2016. Multi-objective optimization based on an improved cross-entropy method. A case of study of a micro-scale manufacturing process. *Information Sciences*.

BIETHAN, J. & NISSEN, V. 1995. Evolutionary algorithms in management applications.

COELLO, RUDNICK & CHRISTIANSEN 1995. Using genetic algorithms for optimal design of trusses.

COELLO, C. 2012. Constraint-Handling techniques used with evolutionary algorithms. Mexico.

COELLO, C. & CHRISTIANSEN, C. 2000. Multiobjective optimization of trusses using genetic algorthms.

COELLO, C., LAMONT, G. & VANVELDHUIZEN, D. 2002. *Evolutionary Algorithms for Solving Multi-Objective Problems*.

COELLO, C. & ZALAZAR, M. 2003. MOPSO: A proposal for multiple objective particle swarm optimization.

CHAVEZ, A. 2018. *Optimización estructural de armaduras planas empleando técnicas heurísticas (Algoritmos Genéticos).* Diploma, Universidad de Matanzas.

CHEN, X. 2013. Learnig with sparsity: Structures, Optimization and Applications. *School of Computer Science Carnegie Mellon University*.

CHEN, Y. & ZOU, X. 2014. Runtime analysis of a multi-objective evolutionary algorithm for obtaining finite approximations of Pareto fronts.

CHENG, J. 2010. Optimum design of steel truss arch bridges using a hybrid genetic algorithm. *Constructional Steel Research*.

DEB, K., THIELE, L., LAUMANNS, M. & ZITZLER, E. 2001. Scalable Test Problems for Evolutionary Multi-Objective Optimization. *IEEE*.

DERRAC, J., GARCIA, S., MOLINA, D. & HERRERA, F. 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*.

DESHPANDE, S., WATSON, L. & CANFIELD, R. 2013. Pareto Front Approximation Using A Hybrid Approch. *International Conference on Computational Science.*

GHARARI, R., POURSALEHI, N., ABBASI, M. & AGHAIE, M. 2016. Implementation of strength Pareto evolutionary algorithm IIin the multiobjective burnable poison placement optimization of KWU pressurized water reactor. *Nuclear Engineering and Technology*.

GIAGKIOZIS, I., PURSHOUSE, R. & FLEMING, P. 2014. Generalized decomposition and cross entropy methods for many-objective optimization. *Information Sciences,* 284.

GOLDBERG, D. 1989. Genetic Algorithms in Search, Optimization and Machine Learning.

GONG, D., WANG, G., SUN, X. & HAN, Y. 2014. A set-based genetic algorithm for solving the many-objective optimization problem. *Springer*.

GUTIERREZ, N., PENICHE, R., HERRERA, G., ALVARADO, R. & CARRION, F. 2012. A Non-Symmetrical Soltuion Applying a Genetic Algorithm with Natural Crossover for the Structural Optimization of Truss Structures. *Ingeniería Investigación y Tecnología,* XIII.

HABER, R., BERUVIDES, G., QUIZA, R. & HERNANDEZ, A. 2017. A simple multi-objective optimization based on the cross-entropy method. *IEEE*.

HOLLAND, J. 1975. Adaptation in Natural and Artificial Systems. *The University of Michigan Press*.

HUBAND, S., HINGSTON, P., BARONE, L. & WHILE, L. 2006. A Review of Multi-objective Test Problems and a Scalable Test Problem Toolkit *IEEE,* 10.

HULTMAN, M. 2010. *Weight optimization of steel trusses by a genetic algorithm.* Master, Lund Universit.

KE, L., ZHANG, Q. & BATTITI, R. 2013. MOEA/D-ACO: A Multiobjective Evolutionary Algorithm Using Decomposition and Ant Colony *IEEE*.

KONA, A., COIT, D. & SMITH, A. 2006. Multi-objective optimization using genetic algorithms: A tutorial.

KOZIEL, S. & YANG, X. 2011. Computational Optimization, Methods and Algorithms, Typeset & Cover Design.

LI, M., YANG, S., LIU, X. & WANG, K. 2013. IPESA-II: Improved Pareto Envelope-Based Selection Algorithm II. *In:* PURSHOUSE, R., FLEMING, P., FONSECA, C., GRECO, S. & SHAW, J. (eds.) *International Conference on Evolutionary Multi-Criterion Optimization.* UK.

MACDONALD, R. 2005. *Genetic Algorithms with Steel Structures a Literatura Review.* Master, Brigham Young University.

MARLER, T. & ARORA, J. 2004. Survey of multi-objective optimization methods for engineering. *Struct Multidisc Optim*.

PEDERSEN, M. & CHIPPERFIELD, A. 2010. Simplifying Particle Swarm Optimization. *Applied Soft Computing,* 10.

PIRES, J. 2014. *Structural Optimization using the Finite Element Method.* Universidad Catolica Portuguesa.

SAKA, M. & GREEM, Z. 2012. Mathematical and metaheuristic application in design optimization of steel frame structures: An extensive review.

SMITH, J., HODGINS, J., OPPENHEIM, I. & WITKIN, A. 2002. Creating Models of Truss Structures with Optimization.

VAMPLEW, P., DAZELEY, R., BERRY, A., ISSABEKOV, R. & DEKKER, E. 2010. Empirical evaluation methods for multiobjective reinforcement learning algorithms.

VIDAL, A. 2013. *Algoritmos Heurísticos en optimización.* Maaster, Universidad de Santiago de Compostela.

ZITZLER, E., LAUMANNS, M. & THIELE, L. 2001. SPEA2: Improving the strength pareto evolutionary algorithm. *ETHzurich*.