Estudio del problema de programación de turnos de enfermería

Christian Camilo Meneses Pico 2110960

Trabajo de Grado para optar por el título de Ingeniero Industrial

Director

Javier Eduardo Arias Osorio

Magíster en Administración

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Estudios Industriales y Empresariales

Bucaramanga

2017

AGRADECIMIENTOS

A la razón de poder seguir adelante a pesar de todo: mis padres Martha Pico, Jaime Meneses y mi hermano Jaime Andrés Meneses… les debo todo lo que soy.

A mi novia Maria Alejandra, por acompañarme en este camino y brindarme todo su apoyo, cariño y comprensión.

Finalmente a la UIS y a todos los que hicieron parte de mi proceso formativo y contribuyeron en hacerme una mejor persona.

Tabla de contenido

Lista de figuras:

Lista de tablas

Lista de apéndices:

RESUMEN

TÍTULO: ESTUDIO DEL PROBLEMA DE PROGRAMACIÓN DE TURNOS DE ENFERMERÍA[*]

AUTOR: CHRISTIAN CAMILO MENESES PICO[**]

PALABRAS CLAVE: INVESTIGACIÓN DE OPERACIONES, PROBLEMA DE PROGRAMACIÓN DE TURNOS DE ENFERMERÍA, PROGRAMACIÓN DE TURNOS DE TRABAJO.

DESCRIPCIÓN:

El problema de programación de turnos de enfermería (NSP) es un problema multiobjetivo complejo al cual se le ha prestado especial atención en la investigación reciente. La necesidad de mantener al personal satisfecho en su trabajo toma cada vez mayor importancia en el sector salud debido a su impacto directo en la satisfacción de los usuarios del servicio; es por esto que es cada vez mayor la necesidad de programar horarios que reflejen las preferencias de los empleados a la vez que estos horarios sean eficientes en términos de cumplimiento de demanda. En el presente trabajo se desarrolla un modelo para el NSP basado en auto-programación, los resultados muestran que es factible utilizar este tipo de modelo para dar solución al NSP y se obtienen grandes ventajas con respecto al modelo clásico de formulación de NSP. El modelo se resuelve utilizando dos técnicas modernas de solución: La búsqueda adaptativa en el vecindario (ANS) y el algoritmo híbrido de colonia de abejas (HABC) logran ser eficaces para dar solución al problema formulado con auto-programación. Los resultados de este trabajo muestran que es posible utilizar modelos como la auto-programación para representar más verazmente al problema a la vez que satisfacer las necesidades de demanda crecientes en el entorno actual.

---

[*] Tesis de pregrado

[**] Facultad de Ingenierías Fisicomecánicas. Escuela de Estudios Industriales y Empresariales. Director: Christian Camilo Meneses Pico

ABSTRACT

TITLE: STUDY OF THE NURSE SCHEDULING PROBLEM[*]

AUTHOR: CHRISTIAN CAMILO MENESES PICO[**]

KEYWORDS: OPERATIONS RESEARCH, NURSE SCHEUDLING PROBLEM, SHIFT SCHEDULING.

DESCRIPTION:

Nurse Scheduling Problem (NSP) is a complex multiobjective problem wich has received special attention on recent investigations. The need of keeping the staff satisfied in their job takes more importance in the health sector; this is why its important to schedule timetables that reflect the preferences of the employees and also be efficient in terms of demand. In this work a model for NSP is developed based on auto-scheduling, the results show that it is possible to use this type of model for solving the NSP and obtain big advantages over the classical formulation methods. The model is solved using two modern techniques of solution: The adaptative neighborhood search (ANS) and the hybrid of the bee colony algorithm (ABC) show to be effective for solving the problem formulated with auto-scheduling. The results of this work show that it is possible to use models like the auto-scheduling model to represent more truly the problem and at the same time satisfy the bigger current demand of staff.

---

[*] Bachelor Thesis

[**] Facultad de Ingenierías Fisicomecánicas. Escuela de Estudios Industriales y Empresariales. Director: Christian Camilo Meneses Pico

Introducción

Introducción:

Las necesidades de los seres humanos crecen exponencialmente, los avances en salud del último siglo han permitido un crecimiento y una longevidad mayores que en ninguna otra época para el ser humano. Sin embargo, este desarrollo debe ir acompañado de una adecuada planificación en la que múltiples disciplinas contribuyan a crear un entorno habitable adecuado, sostenible e incluyente. Las necesidades de salud requieren a su vez un sistema que ofrezca servicios de manera eficiente. Una problemática abordada de manera constante en la literatura es la programación de turnos de personal de diferentes sectores productores o de servicios; estos problemas consisten en asignar actividades de la manera más eficiente posible al recurso humano presente en la organización. Múltiples modelos, especialmente de programación lineal, se han desarrollado para resolverlos con éxito y mejorando los resultados respecto a maneras tradicionales anteriormente usadas para programar el personal. Sin embargo, existen sectores en los que su complejidad y gran tamaño hacen que los modelos desarrollados para programación de personal no sean aptos para resolver sus necesidades. Un ejemplo de esto es la programación de turnos de enfermería en el sector salud, tema del presente proyecto; en estos problemas se utiliza generalmente la programación manual, lo cual conlleva al no cumplimiento de los objetivos del hospital con respecto al personal, a un personal insatisfecho y a una afectación del nivel de servicio para el usuario.

Justificación del proyecto

La programación de enfermeras se relaciona con otros recursos propios de la atención médica y su rol es bastante importante en la provisión de un servicio médico de alta calidad, siendo usualmente las de mayor interacción con los usuarios de este servicio. Una programación deficiente resultará en insatisfacción laboral y una alta rotación de personal, especialmente en países en vía de desarrollo, la migración de enfermeras hacia otros países con mejores posibilidades hace que el sistema de salud del país en cuestión se vea afectado y exista escasez de talentos (M'Hallah y Alkhabbaz, 2013). Adicionalmente la insatisfacción laboral en prestación de servicios de salud es una consecuencia indeseada debido a los altos riesgos de que existan errores humanos, así como el deterioro del servicio en general. Los horarios atípicos y en muchas ocasiones largos a los que deben someterse las enfermeras es una característica que debe ser tratada con especial atención.

El propósito de esta investigación radica en mejorar la forma de solucionar el problema de programación de turnos de enfermería mediante un método efectivo y cercano a la realidad actual del problema en cuestión. En base a una revisión detallada de la literatura sobre el tema se propondrá un modelo de solución que busque aumentar la calidad de la programación y hacer uso de la menor cantidad de recursos.

Planteamiento del problema

Un problema de programación de turnos de enfermería consiste en asignar una serie de recursos, en este caso enfermeras, a un conjunto de turnos especificados para un periodo definido. El problema está sujeto a una serie de restricciones tanto duras como blandas y la naturaleza del problema lo hace un problema de tipo NP-hard cuyos objetivos usualmente

son múltiples. La razón por la que existe una alta producción en la literatura científica de este tema es por la complejidad del problema y la variación de los parámetros del mismo entre diferentes instancias.

Objetivos

Objetivo general:

Desarrollar un estudio para abordar el problema de turnos de enfermería desde la optimización matemática.

Objetivos específicos:

- Realizar una revisión de literatura sobre los modelos de optimización y técnicas aplicadas al problema de programación de turnos de enfermería.
- Formular el modelo matemático a trabajar y definir las técnicas de optimización a utilizar.
- Desarrollar y evaluar los algoritmos en Matlab que permitan probar las técnicas seleccionadas sobre instancias encontradas en la literatura.
- Elaborar un artículo publicable sobre el tema investigado.

Metodología

Revisión de la literatura: Con el fin de obtener información sobre los trabajos realizados en el entorno académico sobre el tema de Programación de turnos de enfermería se procede a realizar una revisión de la literatura sobre este tema. De acuerdo a un análisis preliminar de algunos artículos representativos del tema, se construyó la siguiente ecuación de búsqueda para aplicarse en la revisión literaria; la búsqueda fue realizada en las bases de datos electrónicas de la Universidad Industrial de Santander (Scopus, Science Direct entre otras):

Ecuación de búsqueda 1:

*TS= ((Nurse and (Scheduling or Timetable or Planning or Reprogramming or Shift$ or *staff* or *roster*) and Cost$ and (*Optimiz* or "Model Optimization" or Model or method$))) and TI= ((Nurse or Staff or Personel) and (Scheduling or Timetable or Planning or Reprogramming or Shift$ or *roster*))*

Se aplica inicialmente una ventana de tiempo a partir del año 2000 y se modifica la ecuación de búsqueda 1 como sigue:

Los términos *Cost*, *Personel* y *Method* fueron suprimidos y se añadió a la ecuación *Shift* y *Modeling*.

Ecuación de búsqueda 2:

(TS= ((Nurse AND (Scheduling OR Timetable OR Planning OR Reprogramming OR Shift OR staff* OR roster*) AND (Optimiz* OR "Model Optimization" OR Model*))) AND TI= ((Nurse OR Staff) AND (Scheduling OR Timetable OR*

Planning OR Reprogramming OR Shift OR roster))) AND (stype.exact("Scholarly Journals")*

Con esta última ecuación de búsqueda se obtuvo 842 resultados.

Finalmente en una nueva búsqueda, utilizando la ecuación 2, la ventana de tiempo fue reducida para artículos publicados a partir del año 2010-2016 y se obtuvo un total de 521 resultados. Se eliminaron las repeticiones exactas para obtener un total final de 271 resultados.

De acuerdo a los resultados finales, se procedió a seleccionar solamente las publicaciones relacionadas con el tema de Programación de turnos de enfermería, excluyendo los resultados que no tuvieran que ver con esta temática. Como resultado de esta etapa se seleccionaron finalmente 85 resultados.

Selección del modelo y las técnicas de optimización: De acuerdo a una revisión representativa de la literatura se diseña un modelo de optimización del problema de manera tal que se represente lo más realista posible el problema de programación de enfermeras; para esto se tiene en cuenta aspectos clave mencionados por los autores en la literatura como influyentes en la precisión y robustez del modelo, proceso de selección de parámetros, representación acertada de las preferencias humanas y del entorno social y operacional del problema. En base a la fase de selección del modelo y tomando la revisión de la literatura como referencia se selecciona una técnica de optimización efectiva para dar solución al modelo planteado anteriormente; esta técnica de optimización se escoge teniendo en cuenta los diversos estudios y aplicaciones que han realizado los investigadores en esta área para finalmente seleccionar la más apta y efectiva que solucione el modelo planteado.

Experimentación

El modelo y por consiguiente la técnica de optimización escogida serán puestos a prueba en un software computacional que recree los parámetros y variables escogidas para la representación del problema. Se comparará con otras técnicas actuales en el mismo contexto y se comprobará su efectividad para representar y dar solución al problema de programación de enfermeras de manera acertada y efectiva.

Elaboración de un artículo de investigación

Se desarrollará un artículo de carácter publicable con el estudio y todos sus componentes.

Tabla de cumplimiento de objetivos

Realizar una revisión de literatura sobre los modelos de optimización y técnicas aplicadas al problema de programación de turnos de enfermería.	Capítulo 3
Formular el modelo matemático a trabajar y definir las técnicas de optimización a utilizar.	Capítulo 4
Desarrollar y evaluar los algoritmos en Matlab que permitan probar las técnicas seleccionadas sobre instancias encontradas en la literatura.	Capítulo 6
Elaborar un artículo publicable sobre el tema investigado.	Anexo C

1 Marco teórico

1.1 Investigación de operaciones en el cuidado de la salud:

La investigación de operaciones (IO) ha aportado significativamente al área de la salud, especialmente en las últimas décadas en las que la población ha crecido exponencialmente, han sido necesarias herramientas de optimización y un enfoque a la búsqueda constante de alta calidad en el servicio utilizando de la manera más eficiente posible los recursos. Algo para remarcar es que la simulación y la investigación no-determinística ha ido aumentando en los últimos años (Rais y Viana, 2011).

Las áreas de investigación clave de la IO en el sector incluyen:

1. Planeación del servicio
2. Logística y programación de recursos
3. Prácticas para el cuidado de la salud

1. Planeación del servicio : Dentro de la planeación del servicio el estudio en este tema se puede clasificar en varias categorías:

1.1. Previsión de la demanda: Dada la importancia de analizar la variabilidad de la demanda, varios autores han estudiado este tema y definido modelos cuantitativos para tratarla.

1.2. Localización

1.2.1.1. Localización de instalaciones: Una gran cantidad de autores se han dedicado a estudiar lugares de localización de instalaciones como centros de salud, centros de cuidado preventivo, entre otras. También es objeto de estudio el problema de trasplante de órganos y recolecta de sangre.

1.2.1.2. Vehículos de emergencia: En una menor medida se han realizado estudios sobre la distribución y el manejo de ambulancias o vehículos de emergencia relacionados.

1.3. Planeación de la capacidad: Son varios los estudios sobre planeación de la capacidad dentro de los que se encuentran: atención en sala de urgencias, capacidad de las salas de cirugía, capacidad general en pacientes.

2. Logística y programación de recursos: Estos dos temas son los más estudiados dentro de la literatura, específicamente el tema de programación de enfermería ha sido el más destacado. Dentro de esta temática se encuentran diversas categorías:

2.1. Programación de pacientes

2.2. Programación de recursos: Salas de cirugía, enfermeras, médicos.

2.3. Logística: Relacionado con el manejo óptimo de productos en los almacenes y el flujo efectivo de estos a través de las instalaciones.

3. Prácticas para el cuidado de la salud: Son varios los aportes que han realizado los investigadores de IO para mejorar el cuidado de salud. Dentro de estos aportes se encuentran:

3.1. Diagnóstico de enfermedades: Dentro de este campo destacan los trabajos de prevención de ataques epilépticos, mejoramiento de tratamientos contra el cáncer, el asma y monitoreo de riesgos arteriales, entre muchos otros.

3.2. Planeación de tratamientos: Los estudios en este tema abarcan el tratamiento por radioterapia contra el cáncer, estudiar la frecuencia ideal de exámenes de mamografía, de riñón, procesos de decisión para tratar la braquiterapia (tratamiento contra el cáncer) y la diabetes.

3.3. Donación y trasplante de órganos: Se utiliza teoría de colas y simulación para mejorar la toma de decisiones en esta área.

3.4. Prevención de enfermedades: Estos estudios se relacionan con los procesos de selección de vacunas.

1.2 Programación de turnos de personal

Sin importar el tipo de organización, todas tienen la necesidad de administrar de manera eficiente sus recursos; el recurso humano conforma 2/3 partes del presupuesto total de una compañía y la calidad de la programación del talento humano puede ser crítico a la hora de definir el nivel de calidad de la misma. El problema radica en satisfacer un nivel mínimo de capacidad en cierto momento del día o periodo, con información sobre pronósticos de demanda.

La programación de turnos de personal se da en tres etapas generales: Planeación de personal, diseño de los turnos de trabajo y asignación de turnos de trabajo a personal.

1.2.1 Planeación de personal: La previsión del número de empleados por periodo de tiempo puede variar en exactitud de acuerdo a la actividad que desarrolla la organización: la demanda puede ser totalmente conocida como en el caso de las aerolíneas, donde se sabe con exactitud y cierto margen de tiempo el número de vuelos por día. Puede ser parcialmente prevista, en el caso de algunos departamentos de hospital o plantas de producción donde existe una serie de actividades programadas que están sujetas a cambios sobre la marcha. Finalmente la demanda puede ser totalmente prevista, donde no se puede saber con exactitud los requerimientos de personal con anticipación; como es el caso de una sala de urgencias o un centro de llamadas.

En relación con la demanda, ésta se puede definir para el caso de las empresas de servicios como *"el número de servicios a ser efectuados por la fuerza laboral de una compañía sobre el horizonte de programación"* (Dorne, 2008).

Finalmente, los parámetros que describen el nivel de servicio deseado por una compañía son los siguientes y se expresan en unidades de tiempo, habilidades o área de aplicación.

- Requerimiento mínimo de empleados
- Requerimiento ideal
- Requerimiento máximo

1.2.2 Diseño de los turnos de trabajo El diseño de los turnos de trabajo se considera una decisión de carácter táctico en la mayoría de organizaciones, sin embargo esta clasificación puede variar dependiendo del tipo de actividad de la organización y de su estrategia para enfrentar los cambios de demanda y atender el nivel de servicio. Recientemente ha sido posible incluir como variable de optimización el diseño de los turnos (su duración, periodo, etc.) gracias a las herramientas de simulación que permiten observar el impacto causado por los cambios en determinado turno sobre el sistema. Sin embargo, la mayoría de herramientas de programación se enfocan en asignar empleados a los turnos ya definidos, ya que realizar modificaciones a los turnos suele ser de una mayor complejidad.

Las variables a tener en cuenta en el diseño de los turnos son el personal, los turnos y el horario de trabajo. El personal usualmente es agrupado en unidades o grupos de acuerdo a sus habilidades o demás características, esto hace más posible la programación y el diseño de los turnos de trabajo. Los turnos de trabajo se definen como un intervalo de tiempo donde los empleados son programados para realizar sus actividades; los turnos se pueden replicar cada determinado tiempo y se puede solapar entre sí. Finalmente, el horario de trabajo es el resultado de la combinación agregada de los turnos de trabajo y es aplicable dentro de un horizonte de tiempo dado. Las variables a tener en cuenta para definir los horarios de personal son las siguientes:

- Máxima y mínima cantidad de horas de trabajo y horas de trabajo extras.
- Periodos de descanso entre turnos.
- Días libre, vacaciones.
- Compatibilidad de los turnos con el personal.

- Preferencias individuales.
- Nivel de habilidades de los empleados.
- Nivel de servicio máximo, mínimo y deseado.

Las variables anteriormente mencionadas son una representación general de los aspectos a tener en cuenta a la hora de diseñar y programar los turnos de personal; sin embargo, se deben tener en cuenta también factores tanto internos como externos de la organización como lo son: las características de la fuerza laboral, las características de las actividades a desarrollar en términos de esfuerzo físico, las leyes laborales aplicables en el lugar geográfico de actividad, los tiempos de operación, entre muchos otros. La combinación de todas estas variables junto con el objetivo de maximizar la eficiencia en el uso de los recursos hace de este problema un problema complejo.

1.2.3 Asignación de turnos de trabajo El problema de la asignación de turnos radica en asignar recursos a horarios de trabajo para satisfacer la demanda. Este proceso puede tardar muchas horas si es efectuado de forma manual y entre mayor sea el tamaño de la compañía, mayor tiempo será necesario. Adicionalmente tanto en empresas de manufactura como de servicios, la asignación de turnos de personal se debe realizar de la mejor manera posible, satisfaciendo los niveles mínimos de empleados presentes en cierto periodo, programando factiblemente los turnos y teniendo en cuenta las restricciones en el momento de asignar empleados a turnos creados; todo esto para lograr el objetivo de aumentar el nivel de servicio, con la misma cantidad de recurso humano.

Debido al conflicto entre los objetivos, la calidad de la asignación de turnos es cuestionable si se realiza de forma manual, incluso para los empleados más experimentados

manejar un número tan grande de variables resulta bastante agotador y su optimización imposible.

En esta sección se definió el proceso general de programación de turnos de personal y se evidenciaron las dificultades de realizarlo de forma meramente manual. Cabe resaltar que los factores sociales son igual de importantes que los factores técnicos a la hora de programar los turnos; se debe analizar detenidamente las necesidades y requisitos que plantea la fuerza laboral para cumplir los objetivos de nivel de servicio y eficiencia.

1.3 Programación de turnos de enfermería

El problema de programación de turnos de enfermería (NSP) es un problema de tipo NP-Hard en donde se busca la maximización del nivel de servicio y la satisfacción de los empleados con un presupuesto limitante. Este problema ha recibido especial atención en los últimos años debido a múltiples factores: entre ellos el aumento de la demanda, el alto índice de rotación de personal de enfermería y el alto consumo en recursos que caracteriza la programación de enfermeras con los métodos tradicionales.

1.3.1 Definición del problema. El problema de NSP se formula generalmente utilizando programación entera, donde las variables de decisión suelen ser:

- Cantidad y tipo de enfermera a ser programada en un periodo dado.
- Cantidad y tipo de enfermera de tipo externo a ser programada en un periodo dado.

Estas variables determinan el costo por periodo de la programación de los turnos, y se busca especialmente disminuir al máximo la cantidad de enfermeras a ser programadas de tipo externo. Si bien la función objetivo es sencilla de representar, las características

especiales del sector hacen de este problema un problema complejo, algunas de estas características son:

- Horarios de tipo de continuo (24 horas)
- Turnos de trabajo largos
- Alta fluctuación en la demanda

1.3.2 Restricciones. Para asegurar la satisfacción de los empleados y no disminuir el nivel de servicio, el problema de NSP se caracteriza por tener restricciones duras y blandas.

Restricciones duras

Las restricciones duras son similares a cualquier problema de programación de personal y hacen referencia a la cantidad de horas a trabajar por semana, vacaciones, días libre y demás derechos laborales que tienen los empleados:

- Número de horas límite estipuladas en el contrato
- Días de vacaciones
- Días libre
- Tiempo de descanso entre turnos

Restricciones blandas

Las restricciones blandas generalmente se usan para evitar horarios que favorezcan la desigualdad, que disminuyan el nivel de servicio o que disminuyan el nivel de satisfacción de cada empleado, algunas son:

- Valores mínimos y máximos de cantidad de empleados por turno
- Solicitudes para trabajar en turnos específicos
- Igualdad en la asignación de solicitudes
- Entre otras

Si bien los valores mínimos para la cantidad de empleados son tratados como restricciones blandas, en muchos modelos donde el nivel de servicio es prioritario esta restricción es tratada como dura.

1.3.3 Herramientas de solución. La técnica de solución más utilizada es la programación manual, sin embargo, esto toma varias horas de trabajo y la calidad de la programación depende de la experticia de la persona que lo realice; con la ausencia de un modelo matemático que incluya todos los factores mencionados anteriormente, la programación de turnos de enfermería de forma manual es incapaz de generar una solución de calidad, con las consecuencias que esto conlleva.

Es por esto que se han desarrollado desde la década de 1970 métodos de solución propios de la investigación de operaciones; el desarrollo por parte de investigadores de técnicas de solución para este problema ha permitido, en los casos en que han sido implementados, ahorros y una mejora en el nivel de servicio de las organizaciones prestadoras de salud, así como un aumento de la satisfacción laboral (Ramsey-Coleman, 2012).

1.4 Modelos de optimización

El objetivo de un modelo matemático es representar de la manera más precisa posible un problema, usualmente de la vida real, para darle solución de manera cuantitativa. Está conformado por un sistema de ecuaciones matemáticas relacionadas que describen la esencia del problema. De esta manera si deben tomarse *n* decisiones cuantificables relacionadas entre sí, se representan como variables de decisión, para las que se deben determinar los valores específicos. En consecuencia, la medida de desempeño adecuada se

expresa como una función matemática de estas variables de decisión; a esta función se le denomina función objetivo. Para expresar las limitaciones que pueden llegar a tener las variables de decisión se utilizan funciones llamadas restricciones. Finalmente las constantes incluidas en la función objetivo y las restricciones se denominan parámetros del modelo, estas constantes son características de cada problema a representar y son determinadas de acuerdo a datos u observaciones reales **(**Hillier y Liberman, 2010).

Como conclusión se puede definir un modelo de optimización como el problema de elegir los valores apropiados de las variables de decisión, de forma que se maximice la función objetivo sujeta a las restricciones dadas.

1.5 Programación lineal

La programación lineal es uno de los modelos más utilizados en la IO y su aplicación más común es cuando se desea asignar de la mejor manera posible recursos limitados a actividades que compiten entre sí por ellos **(**Hillier y Liberman, 2010, p-21). La variedad de aplicaciones de este modelo abarca una gran cantidad de problemas y se utiliza en distintos sectores. El adjetivo lineal de este modelo significa que todas las funciones matemáticas que contiene deben ser lineales; a su vez el término programación no debe ser confundido con el término computacional ya que significa planeación. Por lo tanto, la programación lineal significa la planeación de actividades para obtener un resultado óptimo (Hillier y Liberman, 2010, p 30).

1.6 Programación dinámica

La programación dinámica, en contraste con la programación lineal no cuenta con una formulación matemática estándar del problema, sino que se trata de un enfoque más general. Es útil para la toma de decisiones secuenciales e interrelacionadas, proporciona por lo tanto un procedimiento sistemático para determinar la combinación óptima de decisiones. El problema de programación dinámica se puede dividir en etapas que requieren una política de decisión en cada una, cada etapa con cierto número de estados asociados a ella. El efecto de la política de decisión en cada etapa es transformar el estado actual en un estado asociado con la siguiente etapa (Flor, 2015)

Una clave de este modelo es que su estructura va cambiando a medida que se solucionan los subproblemas del problema mayor, para esto utiliza la memorización para recordar estas soluciones de cada etapa y hacer más efectivo el modelo (Universidad de Granada, 2016).

1.6.1 Programación entera. Si las variables de decisión del modelo solo toman valores enteros en un modelo de programación lineal a este se le denomina programación entera; si solamente algunas de las variables toman valores enteros al modelo se le denomina programación entera mixta. Este tipo de modelos es útil para representar variables que involucren cantidades (personas, vehículos etc..), sin embargo, su aplicación más importante es la programación binaria. La programación binaria es una forma de programación entera, ya que existen algunas variables de decisión de tipo *sí o no,* estas se representan mediante 0 y 1. Estos modelos involucran más directamente la toma de decisiones al respecto de un recurso cuando este tiene valores o parámetros establecidos (Hillier y Lieberman, 2010, p. 428-495).

1.6.2 Programación no lineal. En muchos casos los problemas del mundo real no se pueden representar con funciones lineales solamente, en estos casos se utiliza la programación no lineal. La diferencia respecto a la programación lineal es que esta no tiene un método predeterminado para solucionar los problemas y la solución, así como sus métodos y por lo tanto su complejidad depende de la naturaleza del problema. Existen problemas en los que solamente la función objetivo es no lineal y las restricciones son lineales, algunos donde la función objetivo es cuadrática, convexa etc. Si bien no se tiene un método definido de solución para todos los problemas no lineales, su categorización de acuerdo a la naturaleza del problema es bastante útil para aplicar métodos relacionados.

1.7 Métodos de optimización

1.7.1 Método Simplex. El método Simplex fue desarrollado por George Dantzig en 1947, se ha comprobado su gran eficiencia y se utiliza en software de optimización hoy en día; la esencia de su funcionamiento es la resolución de sistemas de ecuaciones algebraícas, sin embargo sus conceptos fundamentales son geométricos. Algunas características de este método así como sus pasos a seguir están explicadas a continuación:

- El método Simplex analiza solo las soluciones en los vértices, es decir las regiones factibles donde se maximiza de cierta manera la función objetivo.
- Es un algoritmo iterativo que va alternando soluciones factibles en los vértices hasta encontrar la mejor.
- La prueba de optimalidad consiste en verificar que ninguna de las pendientes que salen del vértice-solución sean positivas, si efectivamente ninguna lo es se le puede llamar a esa solución la solución óptima.

Cabe mencionar que el método Simplex no es el único método de solución para modelos de programación lineal, el método Simplex-dual y la programación lineal paramétrica son dos métodos basados en el método simplex que utilizan la teoría de la dualidad para el primero y métodos de análisis de sensibilidad para el segundo (Hillier y Lieberman, 2010).

1.7.2 Técnica de la cota superior. En la aplicación del método Simplex a la resolución de problemas lineales la cantidad de restricciones contenidas en el modelo aumentan el tiempo de procesamiento considerablemente, sin embargo las restricciones de no-negatividad afectan de una manera mucho menor al tiempo de procesamiento. Es por esto que esta técnica de solución consiste en convertir las restricciones comunes o funcionales del tipo $Xj <= uj$ en restricciones de no-negatividad al hacer cambio de variable. El modelo se resuelve de manera normal con el método Simplex con la condición de que ninguna de las variables reemplazadas alcance su cota superior (Hillier y Lieberman, 2010, p-264-267).

1.7.3 Método Branch-and-Bound. El método Branch-and-Bound es un método general de búsqueda que se basa en la estrategia de ramificación y que utiliza la técnica de poda para reducir el espacio de búsqueda considerablemente y hacerlo más efectivo. Su aplicación se da inicialmente a partir del *árbol* de soluciones general y a continuación se definen cotas inferiores y/o superiores a subproblemas del problema raíz. Cuando en un nodo o subproblema la cota local es peor que el mejor valor obtenido hasta el momento este nodo es eliminado o *podado*.(Universidad de Granada, s.f.)

1.7.4 Métodos heurísticos. Debido a la necesidad de obtener mayor eficiencia computacional y la existencia de problemas demasiado complejos para ser resueltos en su optimalidad, los métodos heurísticos se utilizan para obtener soluciones factibles de buena calidad pero no necesariamente una solución óptima. Un método heurístico no da garantía de optimalidad pero si es bien diseñado se pueden lograr soluciones de alta calidad cercanas al óptimo, estos métodos se basan en lógicas sencillas y consisten usualmente en algoritmos iterativos que van mejorando la solución en cada iteración. Un método heurístico debe ser desarrollado para cada modelo o problema que pretenda resolver y su funcionamiento depende del grado de adaptabilidad que éste posea con respecto al modelo.

1.7.5 Metaheurísticas. Las técnicas metaheurísticas son métodos de solución que poseen una estructura general y criterios específicos de solución los cuales, a diferencia de los métodos heurísticos, se pueden aplicar a distintos modelos por igual. Las ventajas de las metaheurísticas consisten en una mayor eficiencia a la hora de resolver problemas muchas veces imposibles de resolver o que en su solución óptima consumen demasiados recursos. Estos métodos suelen lograr una solución de alta calidad en tiempos menores que los métodos exactos. Otra característica importante de estos métodos es su habilidad para salir fácilmente de óptimos locales y combinar eficazmente técnicas de intensificación y diversificación en la búsqueda de una solución.

Dentro de las metaheurísticas algunas de las más representativas son (Hillier y Lieberman, 2010, p 563-600).

1. Búsqueda Tabú:

   Estos algoritmos incluyen una rutina de búsqueda local la cual consiste en realizar iteraciones en la búsqueda de la mejor solución; en el comienzo del

algoritmo la rutina solo acepta soluciones mejores que la anterior, sin embargo, después de cierto número de iteraciones se aceptan movimientos dentro de la región de búsqueda que no lleven necesariamente a mejorar la solución previa. Posteriormente al ingresar a una nueva región el algoritmo acepta solamente mejoras y se repite el proceso. Sin embargo, un peligro de este algoritmo es que puede girar continuamente entorno a las mismas soluciones y regresar al óptimo local. Para contrarrestar este defecto se utiliza una lista tabú, la cual se encarga de restringir los nuevos movimientos si estas soluciones ya fueron visitadas recientemente en cierto número de iteraciones atrás.

2. Recocido simulado:

   Se basa en el proceso de templado físico en el cual se somete un material a la fundición, posteriormente, a una temperatura T dada, la energía interna de los átomos del material fluctúa, pero tiende a disminuir; ocasionalmente aumenta con una baja probabilidad. Este proceso se puede representar para solucionar modelos matemáticos de la siguiente manera: Si bien al comienzo de la búsqueda la temperatura T es alta y prácticamente se acepta cualquier movimiento dentro de la región de búsqueda, esta temperatura T va disminuyendo conforme pasan las iteraciones y entre más baja sea menores probabilidades van a haber para aceptar movimientos que empeoren la solución actual. Esta propiedad hace que se acepten paulatinamente soluciones buenas sin perder la diversificación.

3. Algoritmos genéticos:

   Estos algoritmos se basan en la teoría biológica de la evolución formulada por Charles Darwin, las soluciones factibles corresponden a los miembros de una especie, donde la aptitud de cada miembro para sobrevivir se mide por su función

objetivo. Posteriormente estas soluciones factibles tienen hijos los cuales son el resultado de la combinación de varias características de los padres. Usualmente los más aptos sobreviven y transmiten rasgos de la solución a las generaciones posteriores, de esta manera se va mejorando la solución a medida que pasa el tiempo. El criterio de parada del algoritmo puede ser el tiempo de procesamiento, un número fijo de iteraciones consecutivas o cierto número de iteraciones sin mejora alguna. Estos métodos se encuentran dentro de la categoría de algoritmos poblacionales los cuales se diferencian por mantener un conjunto de soluciones activas al tiempo; en vez de una solamente.

Con el paso del tiempo la investigación se ha enfocado en crear nuevas metaheurísticas, mejorar las existentes o hacer combinaciones entre ellas. Una rama de las metaheurísticas explorada ampliamente son las metaheurísticas de tipo poblacional y sus variaciones, desde el algoritmo genético se han creado muchas otras que han resultados eficaces en resolver problemas de IO; entre ellas se encuentran los algoritmos similares de colonia de hormigas y colonia de abejas, búsqueda armónica, búsqueda difusa de la metamorfosis, entre otros. Todas ellas utilizan una población de soluciones que es mejorada iterativamente en distintas fases evolutivas (en la búsqueda de la metamorfosis, la población consiste en una serie de elementos que conforman una solución, y no una serie de soluciones.). Otra rama que ha sido bastante explorada ha sido la búsqueda variable en el vecindario (VNS); estas metaheurísticas consisten en mejorar iterativamente una solución de acuerdo a movimientos en el vecindario de la solución actual que crean una nueva solución, usualmente mejorada. Si bien la búsqueda VNS básica posee un alto nivel de intensificación, su estancamiento en un máximo local es frecuente; es por esto que varios autores han mejorado esta técnica al incluir una mayor diversificación en la búsqueda, sin perder el potencial intensificador. Un

ejemplo de esto es la búsqueda adaptativa en el vecindario (ANS) desarrollada por Lü y Hao (2012). La búsqueda ANS es una técnica robusta que utiliza parámetros capaces de aumentar o disminuir el nivel de diversificación o intensificación, de acuerdo al comportamiento de la búsqueda, el ANS incluye técnicas como la búsqueda tabú y alterna entre tres tipos diferentes de búsqueda.

1.7.6 Métodos híbridos. Debido a la necesidad de perfeccionar los anteriores métodos de solución y tomar las ventajas de cada uno para combinarlas, se crean métodos híbridos entre ellos. Un ejemplo de esta combinación es la aplicación del método Branch-and-Bound y el algoritmo genético a un modelo de solución; usualmente los métodos heurísticos o metaheurísticos son combinados con los métodos exactos en la fase de mejoramiento de las soluciones o en la selección de las ramificaciones, entre otras aplicaciones. Los métodos híbridos suelen representar una mejora para solucionar problemas específicos y son aplicados de acuerdo a la naturaleza del problema a tratar. Cuando se utiliza un método híbrido de solución se sustenta su efectividad con respecto a los métodos aplicados individualmente.

Otro ejemplo reciente de método híbrido de solución es el algoritmo de colonia de abejas (ABC) combinado con la heurística de optimización en colina (HCO). Este método, denominado HABC, modifica el ABC común en la fase en la que cada abeja obrera mejora su solución; en vez de efectuar movimientos de vecindario sencillos como en el ABC, el HABC utiliza la heurística HCO en la fase de las abejas obreras para encontrar un máximo local en cada solución; si esta solución es mejor que la anterior, se reemplaza en la población de soluciones. Esta técnica mejora sustancialmente la capacidad intensificadora del algoritmo ABC, sin embargo aumenta el recurso computacional.

1.7.7 Hiperheurísticas. Una adición a los métodos de solución consiste en predefinir heurísticas de solución en el método y crear un algoritmo por sí mismo que solucione el problema de seleccionar las heurísticas más apropiadas para resolver el problema. Por lo tanto, se manejan dos conjuntos de heurísticas, las primeras son las mencionadas anteriormente que se encargan de dar solución al problema de optimización y las segundas se encargan de seleccionar las heurísticas apropiadas para lograr este objetivo. Si bien es una técnica relativamente más moderna que las demás se ha empleado su lógica para crear métodos nuevos de solución como lo son la inteligencia artificial y el auto-aprendizaje, entre otros.

1.8 Modelado del problema NSP

1.8.1 Descripción del modelo. Un problema de programación de turnos de enfermería consiste en asignar una serie de recursos, en este caso enfermeras, a un conjunto de turnos especificados para un periodo definido. La combinación de estos tres parámetros define la calidad de la programación; sin importar la dinámica de la función objetivo. El problema se puede expresar de la siguiente manera:

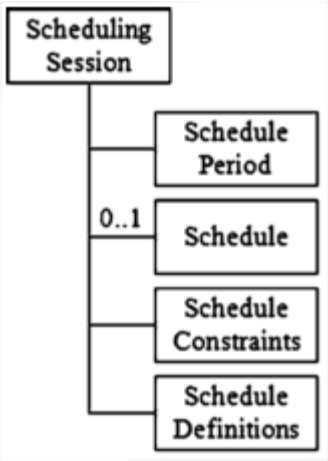


Figura 1: Componentes de un modelo de NSP, fuente: Smet, Bilgin, De Causmaecker y Vanden Berghe (2014, p 3)

En la figura 1 se pueden observar los parámetros presentes en el problema del NSP, en el caso del enfoque del presente trabajo, debido a que se trata de la programación a corto plazo, el único parámetro ajustable es el de **Horario**, los demás se consideran fijos y su modificación depende de políticas de mediano a largo plazo, las cuales se dan en un nivel mayor de planeación y se asumen constantes en el horizonte.

El **Periodo de programación** consiste en el tiempo para el cual se programa un horario específico, usualmente de una semana, por su parte, las **Restricciones de programación** definen las demandas necesarias para cada turno del periodo de programación, así como las habilidades necesarias y el tipo de enfermera requerida para satisfacer esta demanda. Finalmente, las **Definiciones de programación** especifican las características del problema de NSP en cuestión conteniendo información tal como: periodo de vacaciones, tipos de turno, tipo de empleados, restricciones que atañen a los empleados, a los turnos, etc.

Por lo tanto, el problema de NSP se puede expresar como la búsqueda del mejor Horario para un periodo de tiempo especificado, este horario sujeto a las distintas definiciones y restricciones de programación del problema.

1.8.2 Componentes del modelo. A lo largo de la historia sobre el trabajo del NSP varios autores han representado el problema y sus parámetros de diversas maneras:

1.8.2.1 Periodo de programación. El periodo de programación para el NSP puede ser de tipo cíclico o acíclico, en los inicios de la aplicación de Investigación de Operaciones al NSP varios autores definieron el periodo de programación como cíclico con el fin de simplificar el problema y agilizar el tiempo de solución por método enteros. Sin embargo, con el paso del tiempo fueron más los autores que expresaron el periodo como acíclico, manteniéndose esta tendencia hasta el día de hoy; la razón fue que los métodos de solución mejoraron significativamente en rendimiento y esto permitió definir el periodo como independiente a lo largo del horizonte de programación (usualmente de un año). Un periodo cíclico se define como un periodo que se presenta en varias ocasiones a lo largo del año y por lo tanto sus características de demanda y restricciones son las mismas; esto permitía una simplificación del problema, pero una pérdida de exactitud ya que la demanda y las restricciones en el NSP usualmente son cambiantes a lo largo del año y sus patrones son casi imposibles de predecir.

1.8.2.2 Horario. El horario se representa usualmente como una matriz de I enfermeras por D días, de esta manera X(i,d) toma el valor del turno específico si la enfermera i está programada en el día d.

1.8.2.3 Restricciones de programación. Las restricciones de programación son parámetros definidos de acuerdo a las políticas de la organización; estas restricciones abarcan las restricciones de demanda, las cuales especifican la cantidad de enfermeras, el tipo de enfermera, la experiencia que deben poseer, entre otros requisitos, para cubrir un determinado intervalo de tiempo en el periodo de programación.

A su vez incluyen restricciones de colaboración y entrenamiento las cuales limitan la presencia simultánea de tipos de enfermeras en un determinado periodo de tiempo. Finalmente, el horario ya creado de una enfermera puede ser solicitado como "bloqueado" para que la programación final no altere a este horario en particular, a esta restricción se le denomina como restricción de bloqueo.

En este apartado de restricciones de programación la literatura comprende múltiples trabajos para representar los requerimientos de demanda; estas restricciones de demanda pueden ser representadas con un número fijo de empleados requeridos para un turno o intervalo de tiempo dado, y ser tratadas como restricciones duras del problema. Sin embargo, en muchas ocasiones de la vida real estos valores fijos de empleados no se pueden obtener con exactitud, así sean representadas como restricciones duras a ser cumplidas en todo momento. La razón es que el entorno de programación es muy cambiante y el periodo de programación demasiado corto, los empleados pueden ausentarse por diversos motivos y de esta forma impedir el cumplimiento de una cuota exacta de empleados para un turno o intervalo de tiempo. Es por esto que otros autores optan por definir los valores de requerimiento de empleados como un intervalo de mínima y/o máxima cantidad de empleados requeridos; de esta manera es más probable cumplir con un intervalo de valores que con un valor específico. Esta forma de representar la demanda obedece a observaciones del entorno real de programación y es una aproximación más cercana a la realidad del problema.

La decisión de definir las restricciones de demanda como duras o suaves dependen del criterio del programador y se tratará más sobre este tema en el capítulo de restricciones.

Finalmente, además de establecer la cantidad de enfermeras para un intervalo de tiempo, también se debe especificar el tipo de enfermera solicitada, el mínimo nivel de experiencia,

el tipo de habilidad que posea, entre otras características que puede poseer cada enfermera en particular. Estas características adicionales se incluyen para modelar de una mejor manera las políticas del hospital dentro del modelo y hacen también parte de las restricciones de programación.

1.8.2.4 Definiciones de programación. Una vez conocida la demanda (restricciones de programación) y el periodo sobre el cual aplica, las definiciones de programación representan las características de los recursos a programar dentro del NSP. Esto incluye vacaciones, tipos de habilidades, tipos de turno, características de los empleados y un apartado especial para las restricciones que rigen a los empleados, como son el número de horas trabajadas entre muchas otras.

Muchas de estas características han sido tenidas en cuenta desde el inicio del NSP, sin embargo, algunas son más recientes y han ido apareciendo en la literatura para darle mayor exactitud al problema. Es el caso de los tipos de habilidades y el peso que conlleva programar a una enfermera de mayor categoría en un turno que requiere una enfermera de menor categoría. En el caso de los tipos de habilidades se reconoció que, además de tener una categoría que identifica a los empleados, una enfermera puede estar especializada en ciertas tareas o trabajos sobre otros, por lo tanto, es preferible programar a una enfermera con ciertas habilidades en una actividad donde estas sean requeridas. Más recientemente autores (Smet, Bilgin, De Causmaecker y Vanden Berghe, 2014) han agregado la característica del peso para prevenir que una enfermera de una mayor categoría sea programada en un turno que requiere enfermeras de menor categoría.

En este apartado se incluyen las definiciones de las características de los empleados, esto incluye definir las habilidades, el peso, el nivel de experiencia y datos relacionados al

contrato de trabajo, así como preferencias para cada empleado. Al igual que en casos anteriores, una mejora en la representación del modelo se dio con la definición de grupos de datos o sets, ya que varios de los datos planteados en las definiciones de programación comparten características comunes, se hizo útil la definición de grupos de datos para referirse a aquellos que tienen características compartidas, esto con el objetivo de no referirse a estos individualmente y agilizar el proceso.

Finalmente, todas estas definiciones individuales están sujetas a sets de restricciones, las cuales son agrupaciones de restricciones con características comunes que hacen uso de contadores para contabilizar las violaciones de cada restricción realizadas.

1.8.3 Modelo basado en auto-programación. Como se mencionó en el capítulo anterior una serie de restricciones que hay que tener en cuenta en un modelo actual de NSP son los pedidos de los empleados sobre el horario a programar. Actualmente se habla en la literatura de un déficit de enfermeras de alta calidad debido a las malas condiciones de trabajo, especialmente en países subdesarrollados, las cuales emigran en busca de mejores condiciones hacia otros países. Un problema común sin importar el país es también la alta rotación que sufre el personal de enfermería en centros de salud, es por esto que la calidad del horario y la necesidad de satisfacer los requerimientos de las enfermeras se hace cada vez más importante.

El modelo de programación de tipo auto-programación es un método en el cual el empleado es partícipe de cierta manera en la formación de su propio horario de trabajo al incluir su horario ideal de forma parcial o total. Esto permite expresar un gran número de restricciones de una manera mucho más sencilla y adaptativa.

1.8.3.1 Restricciones individuales vs auto-programación. Al momento de formular un problema de NSP, incluir en el modelo las preferencias de las enfermeras con respecto al horario para el que van a ser programadas puede ser una tarea tediosa, subjetiva y, más importante aún, excluyente en el sentido en que no todas las enfermeras verán reflejadas sus preferencias en el resultado del proceso. Si bien para el modelo o algoritmo programador todas las restricciones son tratadas por igual, es evidente que las restricciones que intentan condensar preferencias individuales en una regla general pueden afectar la satisfacción de las enfermeras al sentirse excluidas del proceso. Otro punto por destacar en el modelo tradicional de definir las preferencias es que puede surgir un gran número de restricciones, lo cual hace más pesado al modelo.

Restricciones de este tipo pueden ser las de series de turno indeseadas, días o turnos indeseados, la preferencia de trabajar varios días seguidos, desear no tener día de descanso antes de vacaciones, no desear trabajar en las noches o fines de semana, entre otras. Todas estas restricciones se definen basadas en opiniones mayoritarias o a consideración del propio modelador.

Este método reduce significativamente la complejidad y la cantidad de restricciones relacionadas con preferencias individuales; es por estas razones y por la naturaleza lógica diversa de los empleados a programar que el método de auto-programación es tenido en cuenta en el NSP. Más aún si se resalta que el hecho de incluir al empleado en la programación de su propio horario conlleva a aumentar la satisfacción laboral de éste.

1.8.3.2 Restricciones grupales. Además de tener en cuenta las preferencias individuales, un parámetro a tener en cuenta en el modelo es el de la medición de la igualdad de la programación. Ya sea por el modo tradicional de definir las restricciones o incluyendo la auto-programación, el horario resultante además de abogar por la menor cantidad de violaciones de restricciones individuales, deberá ser equitativo al repartir estas violaciones obtenidas entre todos los empleados por igual. Otro tipo de restricción grupal es la definida por el contrato laboral, consideraciones legales de trabajo y políticas del hospital respecto a la cantidad de horas trabajadas.

1.8.4 Función objetivo. La función objetivo de todo problema de NSP será disminuir en la mayor cantidad las restricciones suaves, sujeta al cumplimiento de las restricciones duras.

1.8.5 Restricciones duras. Las restricciones duras en ciertos casos están relacionadas solamente con satisfacer la demanda en los intervalos de tiempo; otros autores incluyen también la cantidad de horas trabajadas debajo de un máximo.

1.8.5.1 Carga laboral. Ya que por definición las restricciones duras se deben cumplir en todo momento, en este apartado es conveniente asignar los aspectos legales y aquellas políticas clave del hospital. Un factor importante en el sector de la salud es la satisfacción del paciente y evidentemente su seguridad, es por esto que las horas trabajadas por los empleados en un intervalo de tiempo deben estar por debajo de un límite máximo para evitar riesgos de error humano y aumentar tanto la calidad del servicio como la satisfacción del personal; a su vez un hospital que aplique estas políticas tendrá una mejor imagen frente a la sociedad. Adicionalmente se debe dar cumplimiento a los aspectos legales del contrato que incluye periodo de vacaciones y días de descanso acumulados. Por lo tanto, definidas las restricciones duras como la carga laboral máxima asignada y el cumplimiento de aspectos legales del contrato como vacaciones y días de descanso acumulados, las restricciones suaves se definen como el horario y la calidad con la que éste es programado.

1.8.6 Restricciones suaves. Dentro de las restricciones suaves se encuentran aspectos mencionados anteriormente como el cumplimiento de la demanda de empleados en un intervalo dado, las preferencias individuales y la igualdad en la programación.

1.8.6.1 Restricciones de demanda. La razón de incluir restricciones de demanda como duras o suaves depende de la política del hospital, sin embargo, una estrategia viable es asignar un valor promedio a cumplir para la cantidad de empleados requeridos y penalizar exponencialmente la restricción conforme se aleje de este valor. Ya que la cantidad de empleados reales programados puede ser menor o mayor, el hospital puede considerar más importante prevenir la escasez o el exceso y configurar los parámetros de la penalización a su gusto. Esta restricción considerada como suave debe ser la de mayor prioridad a cumplir sobre las demás de este tipo, de esta manera se asegura una cuota de empleados requeridos con un intervalo de error con tendencia a la baja.

1.8.6.2 Preferencias de los empleados. Las preferencias de los empleados son tratadas como suaves en la literatura, lo cual se adopta en el presente modelo, la razón de que sean restricciones suaves es debido a la imposibilidad de satisfacer todas las preferencias de los empleados al mismo tiempo. La metodología de la auto-programación presenta ventajas considerables respecto a la forma tradicional de definir las restricciones de satisfacción. En la información proporcionada por el empleado sobre su horario (parcial o completo) ideal para el periodo, están implícitamente incluidas sus preferencias y no es necesario formularlas una por una dentro del modelo. La evaluación de estas restricciones se simplifica al medir el nivel de desviación entre el horario propuesto por cada empleado y el horario final obtenido. Más adelante se especificará con mayor detalle la metodología de la auto-programación y su relación con la evaluación de la función objetivo.

1.8.6.3 Igualdad en la programación. Al ser el NSP un problema de asignación de recursos humanos en cuya solución seguramente se violarán algunas preferencias individuales, el equilibrio de estas violaciones debe ser distribuido lo más equitativamente posible entre los empleados.

1.8.7 Definición de las características del modelo.

1.8.7.1 Turnos de trabajo. La naturaleza de los turnos de trabajo es dada principalmente por el hospital; los turnos de trabajo son intervalos de tiempo para los cuales una enfermera está activa dentro del hospital y su duración puede ser fija o variable. Usualmente los turnos tienen duración fija y existen tres tipos de igual longitud: Día, tarde y noche. Un hospital puede tener más o menos cantidad de turnos al día, adicionalmente puede existir el caso en que un turno esté activo simultáneamente con otro. Estos parámetros son definidos por el hospital.

1.8.7.2 Características de las enfermeras. Un hospital puede asignar varias clasificaciones a las enfermeras en función de

- Su rango
- Su nivel de experiencia
- Las habilidades que posea
- El peso que conlleva programarla en exceso

Dependiendo de la complejidad y las necesidades del hospital, este puede definir solamente la clasificación básica por rango y, a medida que la complejidad aumenta, adicionar la experiencia, las habilidades o el peso de programación. Esta forma de clasificar

permite tener en cuenta entornos diversos en los que el hospital requiere utilizar estos datos para modelar sus políticas en el modelo matemático.

1.8.7.3 Tipos de contrato. Las enfermeras pueden poseer un contrato a tiempo completo, tiempo parcial, ser practicantes etc. Adicionalmente una enfermera puede especificar en su contrato que solamente trabaja de día o de noche, lo cual también debe ser modelado.

1.9 Alternativas de solución

1.9.1 Consideraciones sociales. Drake (2014) menciona que existen consideraciones de relaciones humanas y de dinámica de grupos en la programación de turnos de enfermería y que son factores importantes a tener en cuenta en las distintas fases del modelo de solución. Adicionalmente esta es una de las razones por las que la literatura sobre NSP no encaja en la práctica real del sector salud y los algoritmos y modelos planteados por los autores no son recibidos con completa aceptación en la práctica. Por estas razones un modelo de solución, además de cumplir los objetivos de programación y usar la menor cantidad de recursos para lograrlo, también debe ser incluyente de factores sociales y humanos que pueden alterar la representación correcta del modelo.

Mutingi y Mbohwa (2015) también mencionan la importancia de incluir la igualdad de la programación dentro del método de solución, adicionalmente mencionan que las preferencias y las expectativas de los empleados deben ser también consideradas. “Si bien pueden ser imprecisos y conflictivos, estos factores deben ser considerados cuando se construyan horarios de trabajo.” (Mutingi & Mbohwa, 2015).

1.9.2 Auto-programación. La auto-programación es una forma de programación de personal en la que los mismos empleados son los encargados de generar un horario que satisfaga los requerimientos de la organización y que a su vez resuelva los conflictos surgidos entre los empleados.

La auto-programación consta básicamente de dos partes: comienza por recibir los horarios planteados por las enfermeras y posteriormente asigna una enfermera para resolver los conflictos surgidos en el horario final; este proceso es llamado de negociación y es necesario que varias enfermeras se pongan de acuerdo sobre múltiples conflictos. Este método puede ser tedioso a medida que aumenta el tamaño del personal y dedica la mayor parte del tiempo a resolver problemas de intereses entre los mismos empleados, lo cual puede afectar el clima laboral y generar disputas entre las enfermeras. Adicionalmente, pueden no lograrse los objetivos de la organización si se omite la fase de negociaciones, ya que la programación va a tender por suplir las necesidades del personal pero no necesariamente asegurar los requerimientos de demanda de la organización.

Para contrarrestar estas desventajas, la aplicación de técnicas metaheurísticas a este método es una aplicación viable en términos de eficiencia de uso de recursos y se adapta muy bien a los distintos niveles de complejidad del problema.

1.9.3 Auto-programación y metaheurísticas. Una ventaja de esta combinación es que desaparece el proceso largo y complejo de las negociaciones entre enfermeras para definir un horario factible; con el uso de heurísticas sencillas es posible modificar un horario alimentado por la auto-programación y convertirlo en un horario factible de acuerdo a las restricciones duras especificadas (ver capítulo restricciones duras). Esto mejora notablemente la eficiencia del método y reduce la complejidad de las negociaciones humanas y los conflictos que naturalmente aparecen en el proceso para satisfacer este tipo de restricciones. Posteriormente, un enfoque metaheurístico de solución es aplicado para incluir las restricciones suaves en la solución; una evaluación de tipo función de ajuste es aplicada para cada restricción, y esta a su vez es mejorada. Bajo este mismo enfoque se incluye la restricción consistente en alejar lo menos posible el horario final del inicial, en el cuál se expresan claramente los deseos de cada enfermera respecto a su horario de trabajo.

Una vez la solución es mejorada en esta etapa, se obtiene una solución que satisface las restricciones duras utilizando heurísticas sencillas, y satisface y mejora las restricciones suaves utilizando un proceso metaheurístico de solución.

1.9.3.1 Metaheurística evolutiva de la metamorfosis. La metaheurística evolutiva de la metamorfosis fue propuesta por Mutingi y Mbohwa (2015) para tratar el problema del NSP desde un enfoque evolutivo. Este algoritmo se basa en el proceso natural de la metamorfosis que sufren algunos animales, el cual consiste en un cambio físico ocurrido después del nacimiento del animal y controlado por una hormona especial. Adaptado al NSP, el algoritmo se encarga de mejorar constantemente la solución inicial mediante un algoritmo evolutivo, en el cual a diferencia del algoritmo genético, en cada iteración solamente existe una solución y no una población de soluciones. Posteriormente la "hormona de crecimiento" define el criterio de parada.

1.9.3.2 Búsqueda adaptativa. La búsqueda adaptativa en el vecindario (ANS) es una metaheurística propuesta por Lü y Hao (2012) para NSP que evalúa consistentemente el desarrollo del proceso de solución y alterna entre tres tipos de búsqueda de acuerdo al estado en que ésta se encuentre. Esto le permite alternar entre métodos de solución de manera adaptativa e ir mejorando la solución en cada iteración; el algoritmo termina si después de n iteraciones no hay mejora en la solución.

2 Revisión de la literatura

Muchos autores han tratado el problema de NSP desde distintos enfoques de la investigación de operaciones, desde métodos exactos utilizando programación lineal entera, algoritmos branch-and-price, programación de restricciones… entre otros, hasta la aplicación de las conocidas metaheurísticas adaptadas al problema de NSP siendo las más utilizadas el algoritmo genético (GA) y la búsqueda del vecindario variable (VNS). Se ha observado un incremento en los últimos años de los autores que trabajan con técnicas metaheurísticas, métodos híbridos (entre exactos y metaheurísticas) y el mejoramiento de la eficiencia computacional y de calidad de las soluciones para los métodos exactos. También se han introducido las denominadas hiperheurísticas, las cuales utilizan una serie de heurísticas y tienen la cualidad de seleccionar la mejor heurística de solución al problema específico.

2.1 Métodos exactos

Glass y Knight (2010) estudian cuatro instancias de la literatura y revelan problemas conceptuales relacionados con la ausencia de restricciones de continuidad, a continuación desarrollan un método de solución con enfoque MIP que incluye estas restricciones y trata los periodos de programación como periodos de transición y no como entidades unitarias.

Burke y Curtois (2014) modelan el problema de NSP de forma genérica y aplican el método exacto Branch-and-Price utilizando programación dinámica. Formulan el problema de pricing como un problema de ruta más corta con recursos restringidos y lo solucionan utilizando un enfoque de programación dinámica. La ventaja de esta combinación radica en

el aumento de la eficiencia del algoritmo al probar secuencias de turnos iterativamente para cada empleado y descartar las peores. También aplican el algoritmo de eyección en cadena llamado búsqueda aleatoria profunda (VDS), esta se encarga de aumentar el rango de los pequeños movimientos de búsqueda al unirlos a movimientos más grandes. De esta forma el método escapa de mínimos locales. Los autores concluyen que en un entorno real en el que los usuarios están dispuestos a esperar poco tiempo para obtener una solución, los dos algoritmos planteados son bastante competitivos.

Debido a que los valores de satisfacción de las enfermeras, entre otros datos, pueden ser subjetivos o inciertos Topaloglu y Selim (2010) desarrollan un nuevo modelo multi-objetivo con enfoque MIP para el NSP en el que aplican, por primera vez en la literatura, la teoría Fuzzy al problema de programación de enfermeras desarrollando tres modelos Fuzzy de programación de objetivos. Para darle solución al modelo este es transformado en un modelo de programación lineal entera mixta (MLIP).

Utilizando el método exacto branch-and-price, Maenhout y Vanhoucke plantean primero en **(**Maenhout y Vanhoucke, 2010) distintos métodos de ramificación para el método branch-and-price existentes en la literatura, a su vez que procesos para agilizar el tiempo computacional efectuado. Posteriormente Maenhout y Vanhoucke (2013), basados igualmente en un algoritmo de solución branch-and-price, plantean un modelo que alterna entre la fase de planeación y la de programación, incluyendo en el planteamiento del problema lo que ellos llaman "los tres intereses de los accionistas del problema": Satisfacción laboral, calidad del servicio y eficiencia en los costos.

Yilmaz (2012) plantea un modelo matemático de tipo MILP en el que la función objetivo consiste en disminuir el tiempo inactivo de las enfermeras en un periodo de tiempo dado. Ismail y Jenal (2013) plantean el problema como un modelo cíclico que programa el

horario por un año completo, incluyen un método para calcular la cantidad de días que tendrá el ciclo y aplican un enfoque de programación por objetivos 0-1 para resolverlo. La satisfacción de las enfermeras es tratada de una manera diferente en el trabajo de Lin, Kang, Liu y Deng (2014) al clasificar los días libre y los turnos de trabajo en rangos de satisfacción, se le da mayor prioridad a satisfacer las preferencias de turnos deseados o indeseados. Para resolverlo emplean un enfoque de Programación Lineal entera binaria (BILP) donde tienen en cuenta estos rangos y el histórico de turnos deseados o indeseados para cada enfermera, esto último con el fin de organizar por prioridad de programación a las enfermeras. Una programación igualitaria satisfaciendo las restricciones comunes del NSP es obtenida.

Valouxis, Gogos, Goulas, Alefragis y Housos (2012) formulan el problema como un problema de MIP y lo dividen en dos fases. La primera fase consiste en asignar días a enfermeras y la segunda se encarga de asignar turnos específicos en ese día. Este proceso es ejecutado varias veces hasta que el criterio de parada sea alcanzado, para realizar la búsqueda se utilizan heurísticas para identificar las posibles combinaciones y posteriormente es resuelto con métodos exactos. M'Hallah y Alkhabbaz (2013) plantean como función objetivo disminuir la cantidad de enfermeras contratadas por outsourcing, adicionalmente dentro de las restricciones incluyen el deseo de cada enfermera de trabajar en cierto turno específico sobre otros. El problema es resuelto como un problema MIP. Por el lado de la programación de restricciones como método exacto He y Qu (2012) plantean el problema como MIP y utilizan el algoritmo de generación de columnas (CG) para descomponer el problema en sub-problemas. Debido a que CG es parte del método Branch-and-Price, las columnas que se deben generar lo hacen en este estudio por medio de programación de restricciones (CP). Baeklund (2013) plantea un problema de NSP para un

hospital danés en el que incluye bonificaciones en las horas trabajadas para las enfermeras si éstas trabajan los fines de semana o festivos nacionales; de manera que si una enfermera trabaja los festivos o fines de semana, según el modelo ésta ha trabajado más horas que una que no lo hizo. El problema también es resuelto por CP-CG debido a la facilidad para añadir nuevas restricciones. Wright y Mahar (2013) analizan el efecto que tiene la utilización cruzada (*cross utilization*) de enfermeras entre unidades diferentes y concluyen que este modelo centralizado disminuye la cantidad de turnos extra y turnos indeseados entre las enfermeras. Plantean un modelo de programación entera con dos objetivos que es resuelto por métodos exactos. Rönnberg y Larsson (2010) describen el modelo denominado auto-scheduling para la programación de los turnos y desarrollan un modelo que lo automatiza. El modelo planteado expresa las restricciones suaves y duras en forma de votos en contra o pedidos para trabajar en ciertos turnos realizados por las enfermeras. Posteriormente es formulado y resuelto por métodos exactos como un problema MIP en el cual un voto limitado para no trabajar en un turno específico es considerado como restricción dura.

Liang y Turkcan (2015) tratan el problema de asignar pacientes a enfermeras en una clínica oncológica. Plantean un problema multiobjetivo para optimizar costos, carga laboral y tiempos de espera. El problema es resuelto por el método de la “constante epsilor” el cual convierte los objetivos del problema multi-objetivo a restricciones y posteriormente encuentra soluciones no-dominadas cambiando los límites superiores de dichas restricciones hasta que las soluciones alcancen un valor deseado.

2.2 Metaheurísticas

Zhou, Fan y Zeng (2012) modelan el problema empleando un modelo nuevo para el NSP denominado “Set Pair Analysis” (SPA), este consiste en asignar puntajes a las soluciones encontradas dependiendo de qué tan disímiles son respecto a la solución óptima en la que todas las restricciones se cumplen; resuelven el modelo utilizando GA. Leksakul y Phetsawat (2014) presentan un modelo que utiliza simulación para determinar el número de apropiado de enfermeras por turno; la función objetivo contiene dos componentes: disminuir gastos y aumentar la igualdad de trabajo extra. El modelo es resuelto con un método GA. Burke, Curtois, Van Draat, Van Ommeren y Post (2011) utilizan un modelo genérico de NSP y centran su estudio en mejorar la búsqueda local iterativa (ILS). Especifican cómo medir el progreso que hace la búsqueda a lo largo del tiempo: los parámetros para medir esto son el “horizonte de interés” en el que no interesan los valores iniciales de búsqueda sino el progreso que ésta ha realizado, “criterio de reinicio” y “criterio de olvido de reinicio” estos últimos encargados de especificar cuándo parar la medición. Chiaramonte, Cochran y Caswell (2014) utilizan el método de intercambio entre agentes (CNR) para modelar el problema de NSP. En este modelo cada enfermera es una entidad que busca satisfacer al máximo sus preferencias personales “pagando” a cambio con un aporte a la mejora de preferencias de demanda. De esta manera se genera un modelo que busca satisfacer las preferencias de las enfermeras a la vez que las de demanda. Para mejorar el modelo, se combina con una ILS que permite el intercambio entre agentes de asignaciones al horario o “acciones”.

Bilgin, Causmaecker, Rossie y Berghe (2010) trabajan en la formulación de un modelo completo, genérico, para el problema de NSP; el resultado es un modelo del problema

altamente flexible que abarca la mayoría de posibles instancias que se pueden presentar en diversos hospitales. Con esto, es posible definir turnos de trabajo específicos con posibles cruces entre ellos, adicionalmente proponen un enfoque de solución similar a VNS denominado búsqueda de gran vecindario adaptable (ALNS) que utiliza un set se de vecindarios y explora uno a la vez en cada iteración, posteriormente a cada vecindario se le asigna un puntaje de desempeño y se utiliza la técnica estocástica de la "rueda de la ruleta" en cada iteración para seleccionar el vecindario sobre el cual seguir explorando, teniendo mayor probabilidad el de mejor desempeño. Tassopoulos, Beligiannis y Solos (2015) plantean un modelo matemático **y** proponen un enfoque de VNS para su solución consistente en dos fases, la primera que asigna los días de trabajo a las enfermeras y la segunda asigna los turnos específicos. La diferencia con respecto a los demás trabajos de VNS radica en la forma en que las heurísticas, con sus 9 tipos de intercambio de turno, son aplicadas en el algoritmo: los 9 intercambios son aplicados secuencialmente, si no se presenta una mejora se realiza una leve "perturbación" en la búsqueda cambiando dos turnos de enfermera y día entre sí.

Lü y Hao (2012) plantean un modelo matemático de NSP en el cual se utilizan funciones de ajuste para evaluar la satisfacción de cada restricción suave, la función objetivo del modelo siendo la suma de todos los ajustes de las restricciones. Introducen el método ANS para su solución, el cual está inspirado en VNS con dos tipos de búsqueda adicionales. Cada tipo de búsqueda varía en intensidad y diversificación y se alterna entre ellas dependiendo del estado de la búsqueda actualmente. Finalmente si no ha habido mejora en la última *n* iteraciones, la búsqueda se reinicia.

Todorovic y Petrovic (2013) aplican el algoritmo de Colmena de Abejas (ABC) el cual consiste en explorar el rango las posibles soluciones, explorando más las que son

susceptibles de mejora. Adicionalmente los autores aplican un método adicional al ABC el cual se encarga de descartar los movimientos que no presentan mejora alguna.

Awadallah, Bolaji y Al-Betar (2015) introducen un híbrido entre el ABC y el método de búsqueda intensiva de ascensión en colina (HCO) al NSP. El algoritmo tradicional de ABC es modificado en la fase de la búsqueda de soluciones por las abejas obreras, en esta fase se utiliza el método HCO para encontrar el óptimo local de cada solución almacenada en la memoria utilizando 4 tipos distintos de vecindario.

Avob, Hadwan, Nazri y Ahmad (2013) introducen el algoritmo de búsqueda armónica (HSA) al NSP. El algoritmo se inspira en la armonía que existe en una combinación de notas musicales, traducido al NSP consiste en la búsqueda iterativa de buenas soluciones tomadas de una "memoria armónica" que contiene las mejores encontradas, la decisión de analizar las soluciones contenidas en la "memoria armónica" o las que estén por fuera de ella está dada por un factor de ajuste "PAR" que, junto con un componente estocástico, define la calidad de las soluciones de la memoria armónica y por ende su selección para exploración. Awadallah , Khader, Al-Betar y Bolaji (2014) utilizan la metaheurística HSA para solucionar el NSP y analizan varios métodos de selección de memoria armónica entre otros parámetros de la metaheurística.

En el modelo de Wu, Yeh y Lee (2015) el objetivo es optimizar la igualdad en la asignación de turnos, cada uno con una ponderación asignada de acuerdo a las preferencias de las enfermeras. Para solucionarlo utilizan la metaheurística de enjambre de partículas (PSO) la cual es una metaheurística de búsqueda poblacional en la que las soluciones son mejoradas conforme mejor se desempeñan en cada iteración. Los autores dividen los turnos en patrones de trabajo y asignan secuencias lo más similares posibles entre ellos.

Martin, Oueñhadj, Smet, Vanden Berghe y Özcan (2013) desarrollan un modelo de NSP en el que existen cuatro funciones objetivo, todas relacionadas con aumentar la igualdad entre enfermeras. También desarrollan un modelo de solución en el que utilizan una búsqueda cooperativa de agentes para resolver el problema. Esta búsqueda consiste en una serie de "agentes" cada uno con una metaheurística; la búsqueda consiste en que cada uno de estos agentes evalúa horarios con distintos patrones de turno para obtener un resultado. Todos los agentes comparten sus resultados y el que obtenga mejores puntajes en las funciones objetivo envía los patrones de turno que utilizó a los demás agentes para que estos a su vez mejoren el horario propuesto y la búsqueda se repita. Esto permite una búsqueda diversa con distintas metaheurísticas. Burke, Li y Qu (2012) analizan el problema como un problema de generación/alocación; en primera medida en la fase de generación se producen todos los posibles turnos para las enfermeras que cumplan con una serie de restricciones duras relacionadas. Posteriormente, en la fase de alocación se utiliza la metaheurística SWO (Squeaky Wheel Optimization) para crear un horario inicial que satisfaga las restricciones duras para cada una. Para mejorar la solución incluyendo las restricciones suaves, un modelo multi-objetivo es propuesto y éste es desarrollado por la metaheurística de recocido simulado, SA. Finalmente esta última búsqueda arroja una serie de soluciones Pareto-óptimas para cada objetivo las cuales representan los posibles enfoques que se pueden dar de acuerdo a la función objetivo a la que se quiera dar mayor atención. Mutingi y Mbohwa (2015) desarrollan un modelo Fuzzy para el problema de NSP, las restricciones suaves son evaluadas con funciones de ajuste cada una y la función objetivo es el valor mínimo entre todas estas funciones, adicionalmente tratan los requerimientos de demanda como una restricción suave susceptible de ser penalizada gradualmente dentro de un intervalo definido. Para darle solución introducen el algoritmo

de simulación *fuzzy* de metamorfosis (FSM) al NSP; inspirado en el proceso evolutivo biológico que sufren algunos insectos conocido como metamorfosis. Las soluciones son evaluadas con sus funciones de ajuste respectivas y cada violación a las restricciones es medida respecto a una función triangular o una función lineal decreciente; el puntaje de cada solución siendo la función con menor valor. Posteriormente se mejora la solución iterativamente usando heurísticas sencillas de cambio de turnos; la "hormona de crecimiento" especifica qué elementos (enfermeras o días) son escogidos para mejora de acuerdo a un componente estocástico y al valor de su función de ajuste. Finalmente se le da mayor flexibilidad al FSM al ser posible alterar las soluciones de acuerdo a los distintos "pesos" que se le pueden asignar a cada restricción o conjunto de restricciones, esto permite que el usuario pueda elegir entre darle mayor prioridad a satisfacer a las enfermeras o a cumplir los requerimientos de demanda, o asignarle mayor prioridad a una restricción en particular.

2.3 Híbridos

Burke, Li y Qu (2010) formulan el problema como un problema multi-objetivo en el que se aplica una técnica de descomposición entre restricciones duras y blandas que combina la programación entera (IP) y la búsqueda del vecino más cercano (VNS) para su solución. La programación entera es utilizada en un principio para tratar las restricciones duras. Seguido de esto la metaheurística VNS es utilizada para tratar las restricciones blandas restantes.

Tsai y Lee (2010) formulan el problema de NSP de forma que las enfermeras escojan qué turnos libres desean tener en el periodo, debido a que el proceso de negociación en esta etapa es muy complejo, los autores proponen resolverlo por métodos exactos. La siguiente

etapa consiste en asignar los turnos de trabajo y para esto se formula el problema como MILP y se resuelve por medio de GA.

Della Croce & Salassa (2014) formulan un problema IP y para su solución definen cuatro vecindarios de búsqueda, cada problema de cada vecindario es resuelto a su vez por métodos exactos. Los vecindarios van creciendo en tamaño conforme se llega a una mejor solución global, este método es denominado como "mateheurística". Huang, Lin, Lin , Hao y Lim (2014) formulan el problema con tres tipos de turnos en el que la función objetivo consiste en satisfacer lo mejor posible las restricciones suaves, para solucionarlo utilizan un método de partición de restricciones en el que toman las restricciones que hacen más complejo el problema (las restricciones suaves) y las separan de las demás. Esta primera parte es solucionada por métodos exactos y su objetivo es asignar una solución inicial que cumpla las restricciones duras a la siguiente fase; la cual consiste en resolver todo el problema, incluyendo las restricciones suaves, por medio de EA. Burke, Kendall, Li y McCollum (2010) plantean un algoritmo evolutivo, el cual utiliza rankeo estocástico para comparar las soluciones; adicionalmente introducen la hiperheurística de recocido simulado (SAHH) al EA, la cual se encarga de mejorar las soluciones mediante ocho tipos distintos de heurísticas.

2.4 Heurísticas

Brucker, Burke, Curtois, Qu y Berghe (2010) descomponen el problema en dos partes, la primera trata solamente restricciones de "programación" y la segunda parte trata restricciones de "roster" o conjunto de programaciones, con esto logran simplificar el problema al acotar la búsqueda de soluciones a un menor espacio. Adicionalmente los

componentes de la solución consisten en “patrones” o secuencias de turnos. La heurística propuesta consiste en una serie de algoritmos que mejora iterativamente la solución global, finalmente debido a que pueden existir aún secuencias comparativamente malas, se utiliza una heurística fuzzy que se encarga de mejorar los patrones que tengan peor desempeño, mezclado con un componente estocástico. Constantino et al. (2014) formulan el problema como un problema de grafos en el que los vértices representan las enfermeras y los días. Los autores introducen una heurística determinística denominada MAPA, la cual contiene tres fases, la primera soluciona iterativamente los vértices en cada día, la segunda y tercera consisten en mejorar la solución mediante procedimientos de corte y recombinación y redistribución de turnos, respectivamente.

Ásgeirsson (2014) formula el problema de NSP y lo resuelve utilizando heurísticas con una solución inicial formulada por auto-programación. El método consiste en hacer factible la solución inicial propuesta por los mismo empleados, posteriormente se aplica una serie de algoritmos para mejorar la solución, atacando en cada uno una restricción específica. Wong, Xu y Chin (2014) formulan el problema y utilizan la técnica nominal de grupo (NGT) para clasificar por importancia para las enfermeras las restricciones del problema. La solución inicial consiste solamente en definir los turnos fuera de cada enfermera, esta es generada manualmente. Posteriormente se asignan los turnos con una heurística de asignación de turnos; con la solución inicial definida, se utiliza una búsqueda secuencial local (SLS) que se ocupa de mejorar los valores de las restricciones suaves. La diferencia con los algoritmos existentes es que éste es operado fácilmente en Excel y su reprogramación puede ser efectuada fácilmente por los técnicos del hospital.

Legrain, Bouarab y Lahrichi (2015) plantean automatizar el proceso de programación de enfermeras, analizan hospitales en Canadá para recrear éste proceso por medio de

heurísticas implementadas en VBA de Excel. Asimismo, analizan el proceso manual de programación y sugieren algunas mejoras.

2.5 Hiperheurísticas

Una forma de estudiar la información se da por medio de la minería de datos, la cual inicialmente representa la información en matrices de bancos de datos sin importar sus dimensiones, sin embargo debido a que esto puede resultar en pérdida de información, Asta, Özcan y Curtois (2016) utilizan análisis tensorial para resolver el problema de NSP. El análisis tensorial consiste en dividir la información en múltiples categorías o dimensiones y analizar la relación entre cada una de ellas, de esta manera según los autores se puede observar mejor la relevancia y las relaciones que existen en una gran cantidad de datos comparado con la minería clásica de datos. Para resolver el problema de NSP los autores utilizan una hiperheurística con dos tipos de selección de heurísticas, cuyos resultados para cada una son representados por medio de dos tensores; cada uno contiene una serie de matrices de 2 dimensiones apiladas, lo que los convierte en tensores de tercera dimensión. Posteriormente se realiza un análisis tensorial para identificar cual estrategia de selección de heurísticas y a su vez cuál heurística arrojaron los mejores resultados. Esto es posible debido a que los tensores son particionados, la información de sus resultados extraída y comparada con las demás heurísticas y finalmente se clasifican las mejores y se almacena en la memoria para "aprender" en tiempo real sobre qué solución es la correcta. Smet, Bilgin, De Causmaecker y Vanden Berghe (2014) buscan cerrar definitivamente la brecha entre academia y práctica recalcando la necesidad de incluir variables, restricciones

y una función de evaluación realistas en cuanto a los problemas reales. Los autores consideran que los modelos académicos de NSP deben abandonar la ambigüedad, esto lográndose al incorporar restricciones de continuidad. También se deben tener en cuenta las distintas variantes entre problemas e incluso instancias de los mismos.

Para tratar esta cuestión los autores plantean un modelo genérico robusto que incluye las diferentes políticas que un hospital puede aplicar, introducen los conceptos de *dominio* y *grupos* de turnos, días o tipos de enfermeras. Según los autores esto permite manejar con mayor facilidad el problema a la vez que representar al mismo de una manera más contundente y realista. Como método de solución emplean un enfoque hiperheurístico evaluando distintos métodos de selección y criterios de aceptación. Concluyen que lo más importante al definir el método de solución de este tipo recae en el método utilizado como criterio de aceptación, siendo el de mejor desempeño el de recocido simulado (SA) y el gran diluvio (GD).

2.6 Análisis

Son varias las investigaciones efectuadas para acercar el problema de NSP a la realidad; Drake realiza un par de investigaciones significativas analizando las razones de la brecha entre academia y práctica, primero (Drake, 2014) mediante un estudio en un hospital de Malasia, encuentra que existe una relación entre la calidad de la programación y el número de veces que un turno es auto-programado por la enfermera. También especifica las variables que influyen en la "robustez", es decir la capacidad y flexibilidad que posee el método para representar el problema, de la programación electrónica. Posteriormente (Drake, 2014) concluye que el problema de NSP tiene consideraciones políticas y de

dinámica de grupos lo cual hace que las restricciones de cobertura no sean de tipo "*hard*" sino subjetivas, políticas y negociables. También menciona la importancia de enfocar el estudio de OR hacia estas consideraciones para intentar cerrar la brecha entre academia y práctica.

Ramsey-Coleman (2012) realiza un informe de la exitosa implementación de software basado en auto-programación en una red de 3 hospitales para programar turnos de 3000 enfermeras, como resultado se disminuyeron los costos en 2 millones de dólares, especialmente los costos de incentivos y horas extra.

En el trabajo de Petrovic y Vanden Berghe (2012) Los autores expresan la necesidad evidente de cerrar la brecha entre academia y práctica y enfocan su trabajo a introducir criterios con los cuales se pueden comparar métodos de solución para el problema de NSP, para hacer posibles las comparaciones en cuanto a calidad entre métodos bastante diferentes entre sí, como son la inteligencia artificial (AI) y el método de búsqueda tabú; esto permitirá a los implementadores e investigadores tener un mejor criterio de decisión al seleccionarlos

Haspeslagh, De Causmaecker, Schaerf y Stølevik (2014) describen en su artículo la primera competencia internacional sobre programación de turnos de enfermería (2010), mencionan los enfoques utilizados por los finalistas y comparan su desempeño en distintos niveles de tamaño del problema. Si bien los ganadores de la competencia utilizaron un híbrido entre programación matemática y heurística para acotar el espacio de solución; los autores recalcan la robustez del enfoque utilizado por Burke y Curtois (2014) el cual consiste en un método de eyección en cadena con búsqueda aleatoria profunda, este fue aplicado en las instancias de *sprint* con 10 segundos límite y en las instancias de mayor

tamaño utilizaron un algoritmo Branch and Price el cual utilizó el método de eyección en cadena mencionado para generar las columnas.

Adicionalmente los autores se concentran en mencionar la importancia del modelado del problema de NSP, y recalcan en su estudio la eficacia de algunas técnicas y aportes novedosos al campo del NSP.

3 Formulación del modelo

3.1 Características del modelo:

3.1.1 Matriz de turnos. La matriz tiene como variable X_{ij}, donde *i* es la enfermera y *j* es el día del periodo de programación. Si se toma como ejemplo una instancia con tres turnos de trabajo, X_{ij} puede tomar entonces los valores de cada turno:

- Noche, N: Turno de 0:00 a 8:00.
- Mañana, M: Turno de 8:00 a 16:00.
- Tarde, T: Turno de 16:00 a 0:00.
- Fuera, F: Día no programado.

El turno 'F' siempre será definido como turno fuera y su letra será reservada.

3.1.2 Definición de las características del modelo

3.1.2.1 Turnos de trabajo. Siguiendo el mismo ejemplo de cuatro turnos de trabajo posibles

- Noche, N: Turno de 0:00 a 8:00.
- Mañana, M: Turno de 8:00 a 16:00.
- Tarde, T: Turno de 16:00 a 0:00.
- Fuera, F: Día no programado.

3.1.2.2 Características de las enfermeras. Para cada enfermera $i = [1, I]$

- Vector $X\_i(i, d)$ con las preferencias de turnos.
- Vector lógico $V(i, d)$ con los turnos vetados de cambio.

3.1.2.3 Tipos de contrato.

- Intensidad horaria: Completo.

3.2 Restricciones duras

3.2.1 Un turno por día. Todas las enfermeras serán programadas con un turno por día solamente. Esta restricción dura siempre se cumple debido a la forma de representar la matriz y a que X_{ij} siempre toma un solo valor.

3.2.2 Secuencia de turnos prohibida. Una enfermera no podrá trabajar en una secuencia de turnos prohibida. Un ejemplo de secuencia de turnos prohibida, siguiendo el mismo ejemplo anterior puede ser T-N ya que el tiempo de descanso entre los dos no es suficiente (nulo en este caso).

3.2.3 Día de descanso obligatorio cada 7 días. Una enfermera no podrá trabajar más de 6 días consecutivos sin descanso. En este lapso de tiempo debe existir un turno fuera ('F') programado para la enfermera.

3.3 Restricciones suaves

3.3.1 Restricciones de demanda. Las restricciones de demanda se encargan de definir las necesidades del hospital para suplir el nivel de servicio deseado en determinadas horas del día. Utilizan múltiples parámetros relacionados con las características de las enfermeras (sección 4.5.2) y los turnos de trabajo (4.5.3).

Dadas las condiciones realistas del entorno, definir las restricciones de demanda como un valor rígido a ser cumplido puede no ser representativo de la realidad de la programación ya que en algunos casos la restricción dura puede ser violada indiferentemente. Por otro lado, al definirla simplemente como una restricción suave, no se

le agrega al modelo la importancia que tiene satisfacer la demanda en los hospitales. Es por esto que la función de ajuste que mejor represente a las restricciones de demanda debe contener un corto intervalo de desviación admitida, acompañado de una alta ponderación con respecto a las demás funciones. La función que mejor se adapta a estos requisitos es la función linear decreciente (Figura 3), con un valor *a* no muy grande que penalice las violaciones grandes a la restricción. Esto permite darle flexibilidad al problema manteniendo el nivel de servicio dentro de las prioridades del modelo. Las funciones serán dos: Una que mida sub-programación y otra que mida sobre-programación.

3.3.2 Igualdad en la programación. Como se mencionó en la sección 4.4 la igualdad en la programación de turnos debe ser medida y controlada. Para esto se usa una función de ajuste triangular que mide, para cada enfermera, la desviación respecto a la media general de carga laboral en cada periodo.

3.3.3 Preferencias de los empleados. El apartado de las preferencias de los empleados es complejo debido a que no es muy claro cuales turnos son menos deseados de ser cambiados que otros para cada empleado. Es por esto que se debe emplear un mecanismo que permita definir para cierto número de días de cada semana cuales turnos son más deseados de no ser modificados. Adicionalmente es lógico asumir que el empleado deseará un horario final similar al que propuso inicialmente; los parámetros para medir esta similitud son diversos y van desde distancia entre turnos para el mismo día hasta la cantidad de tipos de turnos similares en la semana.

El objetivo será entonces minimizar la variación de turnos respecto al horario inicial y adicionar mayor penalización si el turno cambiado es un turno vetado. Los turnos vetados

serán almacenados inicialmente en un conjunto V para cada periodo, por lo tanto, la función evaluará la cantidad de cambios de turno total realizada por los algoritmos. Adicionalmente se debe procurar satisfacer las preferencias de los empleados de manera equitativa en la matriz de turnos, por lo tanto a esta restricción se le adiciona un componente que mida igualdad (Similar a restricción 4.3.2) en la satisfacción de preferencias.

3.4 Parámetros del modelo

3.4.1 Horario.

- i: Subíndice para enfermeras $[1, I]$
- d: Subíndice para día en el periodo de programación $[1, D]$
- T:Conjunto de tipos de turno de trabajo
- $X(i, d)$: Tipo de turno asignado a la enfermera i en el día de programación d

3.4.2 Demanda.

- $Vmin(p, d)$: Valor de demanda mínima para el turno p en el día $d; P = \{N, M, T\}$
- $Vmax(p, d)$: Valor de demanda máxima para el turno p en el día d; $P = \{N, M, T\}$

3.4.3 Carga laboral.

- $c(i)$: Carga laboral promedio para la enfermera i

3.4.4 Preferencias.

- $\{Vi\}$: Conjunto de turnos vetados para la enfermera i

- $v(i)$: Variación total de turnos para la enfermera i

3.5 Funciones de ajuste para restricciones suaves

La manera de representar las restricciones suaves en el modelo se da por medio de funciones de ajuste (Mutingi y Mbohwa, 2015). Para cada restricción suave existirá una función de ajuste específica que evalúa el nivel de violación de esta restricción; existen dos tipos de funciones de ajuste:

3.5.1 Función de tipo triangular. Esta función se encarga de modelar restricciones para valores específicos con una media *m* deseada y un valor *a* de desviación máxima permitida (Figura 2)

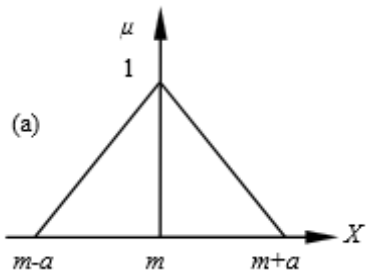


Figura 2: Función de ajuste triangular (fuente: Mutingi y Mbohwa, 2015, p. 4).

3.5.2 Función linear decreciente.

Esta función se utiliza cuando se desea satisfacer un intervalo específico y los valores que no estén dentro del intervalo serán penalizados conforme se alejen más de éste (Figura 3). La diferencia con la función anterior es que en la función linear se acepta una desviación *a* sin penalizar a la restricción, caso contrario a la función triangular, donde la mínima desviación de la media produce una penalización proporcional.

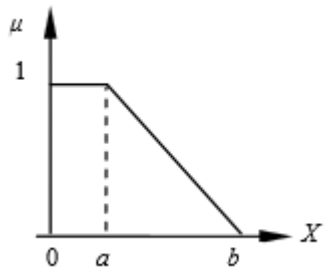


Figura 3: Función de ajuste linear decreciente (fuente: Mutingi y Mbohwa, 2015, p.4.).

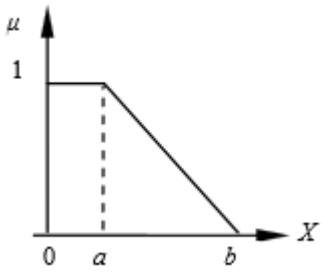
3.6 Definición de las funciones de ajuste

3.6.1 Disminuir la sub-programación. Siendo $Vmin(p,d)$ el valor mínimo de demanda para el turno p en el día d y $Vreal(p,d)$ el valor actual obtenido, entonces

$$X_d = \sum_{p=1}^{P} Vmin(p,d) - Vreal(p,d)$$

La función de ajuste a utilizar es la función linear decreciente

$$\mu_1(Xd)$$



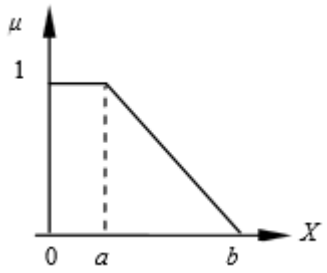
Donde a (faltantes)es la tolerancia para sub-programación y b el límite para el intervalo de demanda

3.6.2 Disminuir la sobre-programación. Siendo $Vmax(p,d)$ el valor máximo de demanda para el turno *p* en el día *d* y $Vreal(p,d)$ el valor actual obtenido, entonces

$$Xd = \sum_{p=1}^{P} Vreal(p,d) - Vmax(p,d)$$

La función de ajuste a utilizar es la función linear decreciente

$$\mu_2(Xd)$$



Donde a es la tolerancia para sobre-programación (sobrantes) y b el límite para el intervalo de demanda

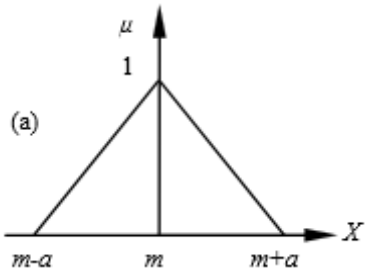
3.6.3 Igualdad en la programación. Siendo *c(i)* el nivel de programación de la enfermera i, entonces

$$Xi = ci - \beta.$$

donde β es el promedio de carga laboral en el periodo.

Se utiliza la función de ajuste triangular

$$\mu_3(Xi)$$



Donde *a* es el intervalo de desviación y m= β

3.6.4 Preferencias de los empleados. Siendo *v(i)* la variación de turnos del empleado *i* :

$$v_i = \beta * V_i + (1 - \beta) * Vp_i$$

Donde β es la penalización por cambiar un turno vetado, V_i es la cantidad de cambios de turnos vetados y Vp_i es la cantidad de veces que un turno normal es cambiado.

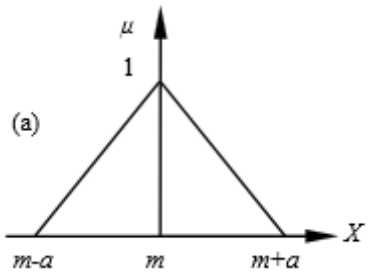
Entonces, la función de ajuste para esta restricción será:

$$\mu_4(Xi) = \mu_{4.1}(Xi_1) * p1 + \mu_{4.2}(Xi_2) * p2$$

Donde

$$Xi_1 = v(i) - \gamma$$

representa la desviación respecto a la media de satisfacción total para todas las enfermeras, siendo γ el promedio para el valor de desviaciones en el periodo. Se utiliza una función de ajuste de tipo triangular para pasar de Xi_1 a $\mu(Xi_1)$:

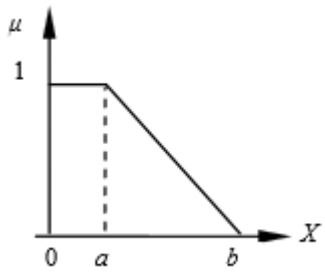


Donde *a* es el porcentaje de desviación respecto a la media (m) permitido y $m = \gamma$.

y

$$Xi_2 = v(i)$$

Xi_2 mide simplemente el nivel de satisfacción de cada enfermera y se utiliza una función de ajuste de tipo linear decreciente para transformar este valor en $\mu(Xi_2)$:



Donde a es el nivel mínimo de insatisfacción y b es el valor máximo aceptado.

Finalmente, ambas funciones de ajuste son ponderadas para dar un valor total a μ_4 . $p1$ y $p2$ siendo los valores de ponderación de $\mu_{4.1}$ y $\mu_{4.2}$, respectivamente.

3.7 Función objetivo

La función objetivo que mide la calidad del horario de programación está definida como sigue:

$$Max(f) = \mu_1 + \mu_2 + \mu_3 + \mu_4 - pen_{H1} - pen_{H2}$$

Donde μ_1, μ_2, μ_3 y μ_4 son las funciones de ajuste para las restricciones suaves de sub-programación, sobre-programación, igualdad de carga laboral y satisfacción de las preferencias. pen_{H1} y pen_{H2} son valores negativos de gran magnitud que miden la violación a la restricción dura #1: secuencia prohibida y a la restricción dura #2: día de descanso.

3.8 Características del NSP

El NSP, sin importar el modelo de representación utilizado, tiene la característica de poseer una gran cantidad de soluciones posibles; esto debido a que es un problema de combinación de recursos limitados; esta característica junto a la naturaleza multi-objetivo del problema en términos de IO, hacen que la solución al NSP raramente satisfaga a todas las restricciones en su totalidad. Es, por lo tanto, labor de los investigadores de IO resolver el NSP de la manera más eficiente, eficaz y efectiva posible para obtener soluciones acorde a la realidad y a las necesidades tanto de empleados como de hospitales y/o centros de salud.

4 Método de solución

Se propone resolver el modelo planteado en la sección cuatro utilizando un método de solución inspirado en la búsqueda adaptativa ANS (Sección 5.1) y posteriormente se resuelve en un marco de HABC (Sección 5.2).

4.1 Búsqueda adaptativa en el vecindario (ANS)

La búsqueda adaptativa alterna entre tres tipos de búsqueda en vecindario con distintas intensidades, el criterio de alternancia entre estas búsquedas es un valor *dl* que incrementa o disminuye dependiendo de si la función objetivo mejora o empeora. El valor de la función

objetivo será entonces la suma de los valores asignados al nivel de satisfacción de cada restricción y el objetivo será aumentar este valor lo máximo posible.

4.1.1 Tipos de movimiento utilizados. Para explorar el vecindario, en cualquiera de las fases se utilizan cuatro tipos posibles de movimientos:

- ρ_1: Cambiar turnos entre dos enfermeras del mismo día, una de ellas con turno Fuera.
- ρ_2: Cambiar turnos entre dos enfermeras del mismo día, ambas con turno ≠ Fuera.
- ρ_3: Cambiar turnos entre dos días para la misma enfermera, un día de estos con turno Fuera.
- ρ_4: Cambiar turnos entre dos días para la misma enfermera, ambos turnos ≠ Fuera.

En cada iteración el movimiento escogido es aleatorio y depende de q

Donde $q = 1 - \varphi * dens$

- Si $random[0,1] < q$ se escoge el movimiento ρ_1 o ρ_3 con una probabilidad de 0.4 y 0.6 respectivamente.
- Si $random[0,1] > q$ se escoge el movimiento ρ_2 o ρ_4 con una probabilidad de 0.4 y 0.6 respectivamente.

Donde $random[0,1]$ es un número aleatorio entre 0 y 1.

4.1.2 Tipos de búsqueda. El parámetro dl controla el tipo de búsqueda que efectúa el algoritmo en una iteración dada, a continuación se mencionan los tres tipos de búsqueda en orden de menor a mayor diversificación y se describe el procedimiento de cada una:

- Búsqueda intensiva: La búsqueda intensiva tiene la particularidad de utilizar lista tabú, debido a que es la que mayor intensificación aporta, se utiliza una lista tabú

para no repetir movimientos y progresar más rápidamente en esta etapa. A continuación se describe el procedimiento:

1. Aplicar movimiento ρ con lista tabú.
2. Lista tabú incluye los movimientos recíprocos por los siguientes $tl + rand * 3$ movimientos. $(tl = 0.8 * E)$ donde E es el número total de enfermeras.
3. Escoge la mejor solución. Si mejor solución es Tabú se escoge siempre y cuando mejore la solución actual.

- Búsqueda intermedia: La búsqueda intermedia se utiliza cuando se desea un nivel de diversificación mayor que la búsqueda intermedia. En esta fase, la búsqueda se limita a la mitad de la matriz X_{ij}, el procedimiento es el siguiente:

  1. Limita la búsqueda a un set $S *$ de enfermeras. $S *= E/2$.
  2. Aplicar movimiento ρ
  3. Escoger la mejor solución.
  4. Si la mejor solución es el movimiento más reciente se escoge la segunda mejor con probabilidad de 0.5.

- Búsqueda diversificada : Finalmente si se desea un nivel mayor de diversificación se utiliza la búsqueda diversificada; esta además de restringir la búsqueda a la mitad de la matriz, escoge un movimiento solamente con el criterio de si mejora o no una sola restricción, sin importar que empeore a las demás. Esta fase aporta una diversificación mayor a la búsqueda ya que, si bien las soluciones en esta etapa pueden no ser mejores, puede ayudar a escapar de mínimos locales.

1. Se escoge un set S* de enfermeras. $S* = E/2$.
2. Se identifican grupos de movimientos sub-prometedores (si por lo menos movimiento mejora una de las restricciones blandas).
3. La solución se escoge al azar entre los movimientos sub-prometedores si los hay. Si no existe ninguno, se escoge cualquier solución del set $S*$.

Criterio de alternancia

El criterio para usar una u otra búsqueda se da con el parámetro dl, el cual mide el nivel de diversificación necesaria en la búsqueda actual.

- Si dl $\in [0, \beta_1)$ →Aplicar búsqueda intensiva.
- Si dl $\in [\beta_1, \beta_2)$ →Aplicar búsqueda intermedia.
- Si dl $\in [\beta_2, 1)$ →Aplicar búsqueda diversificada.

4.1.2.1 Dinámica de dl. Si la función objetivo no ha mejorado en los últimos θ pasos: $\theta = \frac{E*T}{10}$, siendo E el número total de enfermeras y T el número de tipos de turno (sin incluir turno Fuera) el parámetro dl aumenta como sigue:

$$dl \leftarrow dl + \frac{1 - dl}{6}$$

Si por el contrario la función objetivo mejora en cualquiera momento, el parámetro dl disminuye:

$$dl \leftarrow dl - \frac{dl}{10}$$

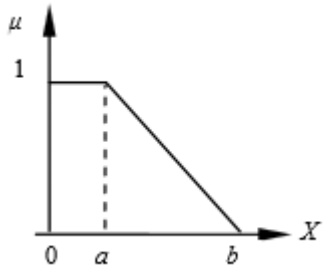
4.1.2.2 Criterio de reinicio. Si no hay mejora en cierto número R de iteraciones se reinicia la búsqueda desde la mejor solución X encontrada o desde la solución X actual, con una probabilidad de 0.5 para cada opción. Adicionalmente el parámetro dl se hace 1 sin importar su valor actual. ($dl = 1$).

4.1.3 Funciones de ajuste utilizadas. Las funciones de ajuste utilizadas para evaluar las soluciones del problema son las descritas en la sección 4.6 (Definición de las funciones de ajuste) del capítulo 4: Formulación del modelo.

4.1.3.1 Restricciones suaves. La función objetivo está compuesta por las funciones de ajuste:

1. Sub-programación:

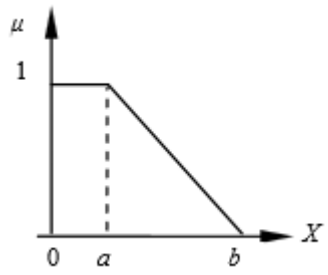
$$\mu_1(Xj)$$



$$X_j = \sum_{p=1}^{P} Vmin(p,d) - Vreal(p,d)$$

2. Sobre-programación:

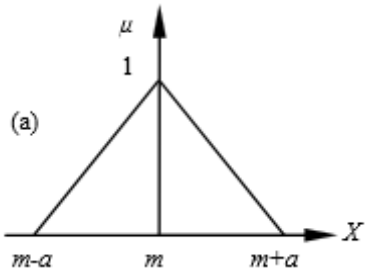
$$\mu_2(Xj)$$



$$Xj = \sum_{p=1}^{P} Vreal(p,d) - Vmax(p,d)$$

3. Igualdad en la programación:

$$\mu_3(Xi)$$



$$Xi = ci - \beta.$$

$$\beta = m$$

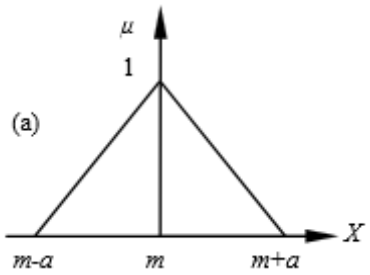
ci: Carga laboral

4. Preferencias de los empleados

$$\mu_4(Xi) = \mu_{4.1}(Xi_1) * p1 \; + \; \mu_{4.2}(Xi_2) * p2$$

$$\text{Con } p1 + p2 = 1$$

$\mu_{4.1}$:

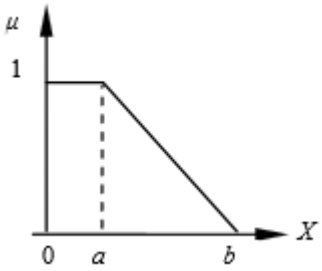


$$Xi_1 = v(i) - \gamma$$

v(i): Variación de turnos

$$\gamma = m$$

$\mu_{4.2}$:



$$Xi_2 = v(i)$$

4.1.3.2 Función objetivo. La función objetivo está especificada tal como se definió en la sección 4.2: Función objetivo:

$$Max(f) = \mu_1 + \mu_2 + \mu_3 + \mu_4 - pen_{H1} - pen_{H2}$$

Donde μ_i son los valores de ajuste para restricciones suaves y pen_{Hi}son las penalizaciones por violar las restricciones duras.

4.2 Búsqueda HABC

La búsqueda HABC fue propuesta inicialmente por (Bolaji, Al-Betar & Awadallah, 2013) .Es un híbrido entre la metaheurística de la búsqueda de colonia de abejas (ABC) y el algoritmo de optimización en colina (HCO).

4.2.1 Algoritmo de colonia de abejas (ABC). El ABC se ha utilizado con éxito en la programación de turnos de personal de diversas áreas (Awadallah, Bolaji & Al-Betar, 2015). Es un algoritmo evolutivo y poblacional que hace uso de operadores no determinísticos para mejorar iterativamente las soluciones presentes en la población, hasta alcanzar un criterio de parada. Las etapas de búsqueda del ABC se asemejan a una población de abejas en su búsqueda por alimento: Existen tres etapas y por lo tanto tres tipos de abejas en ABC: Las abejas obreras, las abejas observadoras y las abejas exploradoras. Cada tipo de abeja tiene una función específica dentro de la búsqueda por la mejor solución.

1. La búsqueda comienza con la generación de una población inicial de soluciones. Se trata de soluciones aleatorias que posteriormente van a ser mejoradas por los tres operadores de la búsqueda (obreras, observadoras y exploradoras).
2. Una vez creada la población inicial, cada solución es modificada por su correspondiente abeja obrera, si el resultado de este movimiento es mejor que el de la solución anterior, esta solución se actualiza dentro de la

población. La operación se realiza de la misma forma para todas las soluciones presentes en la población.

3. La abeja observadora se encarga de escoger una solución, cada una con probabilidad proporcional de ser escogida. Las soluciones peor evaluadas respecto a las demás tienen mayor probabilidad de ser escogidas por la abeja observadora. Una vez escogida esta solución, la abeja observadora efectúa el mismo tipo de movimiento probabilístico que las abejas obreras y si este último resulta en una mejor solución, se actualiza la población con esta nueva solución.
4. Por último la abeja exploradora se encarga de diversificar la búsqueda si alguna solución se encuentra estancada (sin mejora alguna). El criterio para definir cuando una solución se encuentra estancada consiste en el número de iteraciones del algoritmo sin las que haya ocurrido mejora en la solución. Una vez se llega a este valor la función de la abeja exploradora consiste en reemplazar la solución estancada por una solución nueva, aleatoria. Un registro de la mejor solución encontrada por el algoritmo hasta el momento es llevado para evitar la pérdida de información.
5. Los pasos 2,3 y 4 se repiten hasta alcanzar el criterio de parada.
6. La búsqueda se detiene cuando se alcance el número de iteraciones definido o el tiempo máximo especificado como criterio de parada.

4.2.2 Algoritmo híbrido ABC con HCO.

Para mejorar el desempeño de las metaheurísticas, varios híbridos con otras búsquedas han resultado en mayor eficacia, a cambio de un aumento en el tiempo de procesamiento. El

algoritmo de ascensión en colina (HCO) ha demostrado que puede resultar efectivo en su hibridización con otras búsquedas metaheurísticas poblacionales (Awadallah, Bolaji & Al-Betar, 2015); en el caso del ABC y su híbrido con HCO (HABC) éste fue utilizado para resolver el problema de programación de cursos universitarios (Bolaji, Hkader, Al-Betar & Awadallah, 2014) inicialmente y posteriormente aplicado al NSP (Awadallah, Bolaji & AL-Betar, 2015). El objetivo de esta mejora es mejorar la fase intensificadora del ABC al reemplazar el procedimiento común de las abejas obreras por el método HCO.

En este método denominado HABC, las abejas obreras utilizan el HCO para encontrar el óptimo local de cada solución presente en la población. Esto conlleva a mayor uso computacional a cambio de un aumento en la intensificación del algoritmo.

4.2.2.1 Tipos de movimiento. Los movimientos (ρ) utilizados por las abejas obreras y las abejas observadoras son los mismos que los utilizados en la técnica ANS (Ver sección 5.1.1: Tipos de movimientos).

4.2.2.2 Algoritmo HABC. La búsqueda HABC se da como sigue:

1. Iniciación de los parámetros poblacionales:
    - SN: Tamaño de la población
    - MCN: Número máximo de iteraciones
    - $limite$: Número máximo de iteraciones sin mejora, criterio de exploración.
    - HCR: Probabilidad de escogencia para HCO ([0,1])
2. Generación de FSN inicial (Población inicial)
    - Generar diez soluciones iniciales por medio de rankeo de turnos: Esto consiste en ordenar de menor a mayor ‘dificultad’ de programación los turnos a ser programados. El criterio para ordenarlos, es decir la dificultad de cada

turno, se mide por el número total de turnos necesarios en el periodo. Por lo tanto el turno con mayor dificultad es aquel que tenga la menor cantidad de turnos demandados. El método de rankeo de turnos se encarga de crear una solución inicial al asignar enfermeras al azar a los turnos siguiendo su orden de dificultad, comenzando por el que tenga mayor dificultad. El resultado es una solución con enfermeras asignadas al azar a una serie de turnos, 'llenando' primero los de mayor dificultad de programación.

- Ordenar la población inicial de mayor a menor valor calidad de función objetivo.

3. Mejora de cada solución por abejas obreras utilizando HCO.
    - Seleccionar una solución de la población, comenzando por la primera.
    - Comparar parámetro HCR con $random$[0,1]. Si $HCR > random$ efectuar HCO sobre la solución escogida en 1. HCO consiste en efectuar movimiento ρ hasta alcanzar un mínimo local.
    - Pasar a la siguiente solución y repetir paso 2 hasta alcanzar todas las soluciones de la población.
    - Ordenar soluciones de mayor a menor valor calidad de función objetivo.
4. Escogencia por probabilidad proporcional de una solución en FSN para mejora por abeja observadora.
    - Calcular probabilidad acumulada para cada solución de la población.
    - Evaluar la probabilidad acumulada ($prob\_acum$) de cada solución:
    - Escoger la primera solución

- Si 1-$prob\_acum$ <= $random$[0,1] Escoger la solución para mejora por abeja observadora. Si no, continuar hasta encontrar solución que satisfaga desigualdad.
- Aplicar movimiento ρ a solución escogida para mejora.
- Ordenar soluciones de mayor a menor valor calidad de función objetivo.

5. Escogencia de solución estancada para diversificación por abeja exploradora.
   - Se selecciona una solución s al azar entre la población.
   - Se evalúa el número de iteraciones que han pasado desde que solución s ha percibido una mejora ($pruebas$). Este valor se compara con $limite$.
   - Si $pruebas \geq limite$ se reemplaza la solución escogida por una solución aleatoria y se actualiza la población de soluciones. El algoritmo almacena en su memoria la información de la mejor solución encontrada en todo momento.
   - Repetir 2 a 5 hasta alcanzar criterio de parada (MSN).

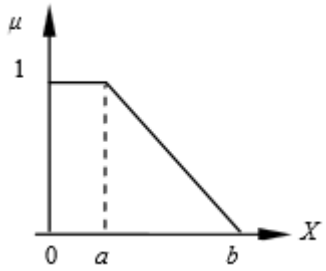
El resultado de la búsqueda es la mejor solución encontrada por el algoritmo.

4.2.2.3 Funciones de ajuste utilizadas. Las funciones de ajuste utilizadas para evaluar las soluciones del problema son las descritas en la sección 4.6 (Definición de las funciones de ajuste) del capítulo 4: Formulación del modelo.

4.2.2.4 Restricciones suaves. La función objetivo está compuesta por las funciones de ajuste:

1. Sub-programación:

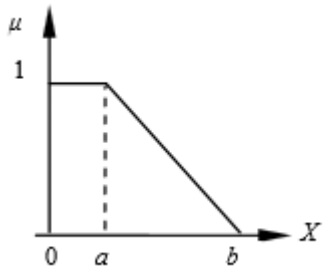
$$\mu_1(Xj)$$



$$X_j = \sum_{p=1}^{P} Vmin(p,d) - Vreal(p,d)$$

2. Sobre-programación:

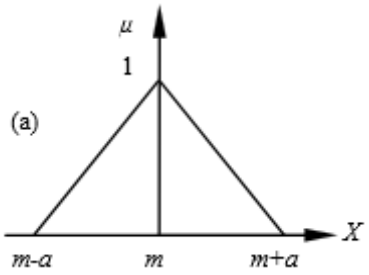
$$\mu_2(Xj)$$



$$Xj = \sum_{p=1}^{P} Vreal(p,d) - Vmax(p,d)$$

3. Igualdad en la programación:

$$\mu_3(Xi)$$



$$Xi = ci - \beta.$$

$$\beta = m$$

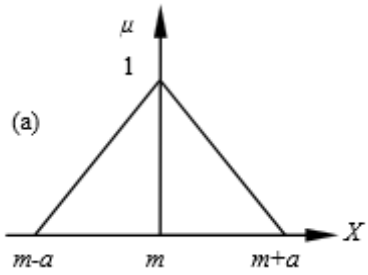
ci: Carga laboral

4. Preferencias de los empleados

$$\mu_4(Xi) = \mu_{4.1}(Xi_1) * p1 \; + \; \mu_{4.2}(Xi_2) * p2$$

Con $p1 + p2 = 1$

$\mu_{4.1}$:

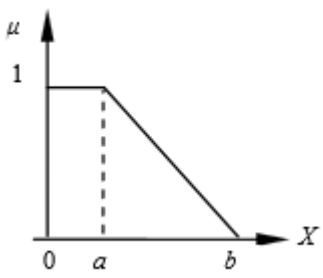


$$Xi_1 = v(i) - \gamma$$

v(i): Variación de turnos

$$\gamma = m$$

$\mu_{4.2}$:



$$Xi_2 = v(i)$$

4.2.2.5 Función objetivo. La función objetivo está especificada tal como se definió en la sección 4.2: Función objetivo:

$$f = \mu_1 + \mu_2 + \mu_3 + \mu_4 - pen_{H1} - pen_{H2}$$

Donde μ_i son los valores de ajuste para restricciones suaves y pen_{Hi}son las penalizaciones por violar las restricciones duras.

5 Implementación de algoritmos en MATLAB

5.1 Implementación del modelo

El modelo planteado en la sección 4 fue implementado en MATLAB.

5.1.1 Valores de entrada. Los valores de entrada necesarios comienzan por la matriz inicial $\boldsymbol{X\_i}$ la cuál es el resultado del proceso de auto-programación propuesto por las enfermeras:

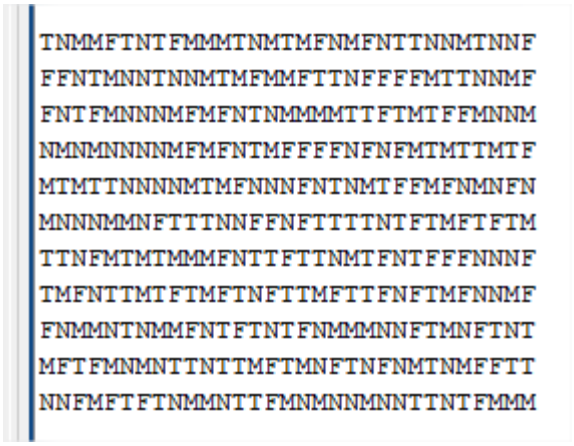


Figura 4: Ejemplo de matriz $X\_i(e,d)$

Los siguientes valores de entrada están relacionados con el nivel de servicio (Nivel de servicio deseado por el hospital) y el nivel de satisfacción de los empleados (En forma de turnos "vetados" por cada enfermera de acuerdo a sus preferencias o jerarquía de turnos deseados). Por lo tanto se pedirá por los siguientes datos:

- Nivel de servicio:
    - $\boldsymbol{V\_min(3,D)}$: Corresponde a una matriz de 3 turnos por D días del periodo de programación y almacena la información relacionada con la cantidad mínima de enfermeras requeridas para un turno en un día específico.

- $\boldsymbol{V\_max(3,D)}$**:** Del mismo tamaño que V_min, ésta almacena la información relacionada con el nivel m áximo deseado de enfermeras en un turno, para un día específico.

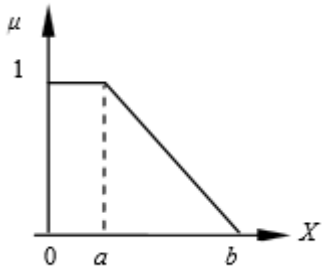
- Satisfacción de los empleados:
  - $\boldsymbol{V(E,D)}$**:** Del mismo tamaño que X_i , es una matriz lógica de turnos vetados de X_i. Cada elemento de V indica 1 si la misma posición en X_i corresponde a un turno vetado para el empleado o 0 si no lo es.

5.1.2 Parámetros de entrada. El cálculo de la función objetivo que refleja la calidad de la matriz de turnos X en cualquier momento está dada como la suma de los ajustes de las cuatro restricciones suaves más la penalización incurrida por las restricciones duras (Ver sección 5). Para las funciones de ajuste; dado que se utilizan varios parámetros para dar el valor de cada una, es necesario leer o asignar éstos de entrada.

Restricción suave de sub-programación

Tal y como se especifica en 4.6.2 es necesario definir los valores para la función de ajuste linear decreciente. Ésta última utiliza los parámetros a y b

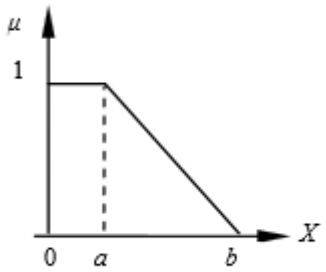
a representa el valor máximo de tolerancia, y b el valor máximo permitido. Para el caso de la función de ajuste para evaluar sub-programación estos parámetros son representados en MATLAB como $\boldsymbol{a\_min}$ y $\boldsymbol{b\_min}$ respectivamente, los cuales son leídos o especificados como parámetros de entrada. En el modelo, a_min y b_min representan la tolerancia para faltantes y el valor máximo de faltantes aceptado, respectivamente. Son leídos o especificados como parámetros de entrada.



Restricción suave de sobre-programación:

Tal y como se especifica en 4.6.3 es necesario definir los valores para la función de ajuste linear decreciente. Ésta última utiliza los parámetros a y b.

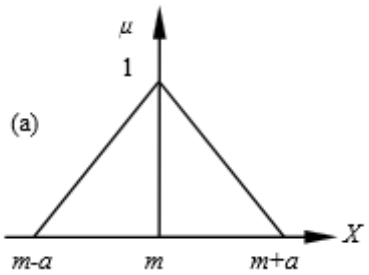
Similar a la restricción suave para sub-programación, a representa el valor máximo de tolerancia de sobrantes y b el valor máximo permitido de sobrantes. Para el caso de la función de ajuste para evaluar sobre-programación estos parámetros son representados en MATLAB como $\boldsymbol{a\_max}$ y $\boldsymbol{b\_max}$ respectivamente, los cuales son leídos o especificados como parámetros de entrada. En el modelo a_max y b_max representan la tolerancia para sobrantes y el valor máximo de sobrantes aceptado, respectivamente. Son leídos o especificados como parámetros de entrada.



Restricción suave de igualdad:

Tal y como se especifica en 4.6.4 es necesario definir los valores para la función de ajuste triangular. Ésta última utiliza el parámetro a.

a representa la desviación máxima permitida respecto a un valor central. En el modelo y para el caso de la función de ajuste de igualdad, el valor central (m) equivale al promedio de carga laboral para todas las enfermeras y a es la desviación máxima permitida (en porcentaje) respecto a la media de carga labora. En MATLAB a es representado como $\boldsymbol{a\_i}$ y es leído a especificado como parámetro de entrada.

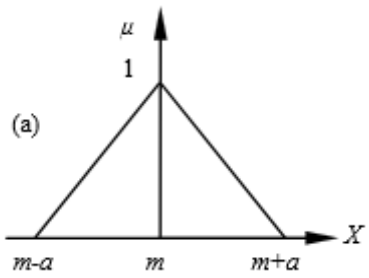


Restricción suave de preferencias de los empleados:

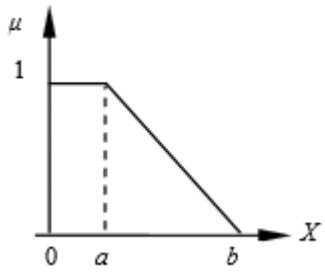
De acuerdo a lo definido en la sección 4.6.5 se necesitan definir los valores para la función de ajuste triangular. Ésta última utiliza el parámetro a.

a representa la desviación máxima permitida respecto a un valor central m. En el modelo el valor central equivale al promedio de violaciones a las preferencias de los turnos programados: Cada turno que es modificado asigna una penalización a esta restricción y cada turno vetado modificado (Matriz $V(e,d)$) asigna una penalización mayor. Por lo tanto

el valor central (m) será el promedio del valor de las penalizaciones por modificación de turnos de las enfermeras y a es la desviación máxima permitida (en porcentaje) respecto a la media de penalizaciones, en MATLAB a es representado como $\boldsymbol{a\_p}$ y es leído o especificado como parámetro de entrada.



Adicionalmente se deben especificar los parámetros para medir la satisfacción global de las enfermeras utilizando una función de ajuste linear decreciente, tal y como se definió en la sección 4.6.5.



a representa el nivel mínimo de insatisfacción y b representa el nivel máximo aceptado. En MATLAB estos parámetros fueron programado como $\boldsymbol{a\_p\_g}$ y $\boldsymbol{b\_p\_g}$ y deben ser leídos o especificados como parámetros de entrada.

Una vez se reciben todos los datos de entrada, es posible asignar un valor a la función objetivo para la matriz de turnos $X(E, D)$ en todo momento. Los valores máximos (ideales) para cada función que componen la función objetivo están resumidos en la siguiente tabla (suponiendo $E = 11$ y $D = 31$ como valores de ejemplo para ilustración).

Tabla 1: Puntaje máximo por restricción

Función	Puntaje máximo
Restr. suave sub-programación	31
Restr. suave sobre-programación	31
Restr. suave igualdad	11
Restr. suave preferencia de turnos	11
Restr. dura #1	0
Restr. dura #2	0

Después de recibida esta información es posible evaluar una matriz $X(e,d)$. La función objetivo está definida como la suma de los valores de ajuste de las restricciones suaves más la penalización sobre la violación de las restricciones duras:

$$f = \mu_1 + \mu_2 + \mu_3 + \mu_4 + pen_{H1} + pen_{H2}$$

5.2 Implementación métodos de solución

5.2.1 Implementación Búsqueda basada en ANS. El método de solución ANS descrito en la sección 5.1 fue implementado en MATLAB (Ver anexo 1: Algoritmo ANS en MATLAB). Se utiliza la formulación del modelo expuesta en el capítulo 4. A continuación se presenta el problema a resolver por el método ANS:

Objetivo: Obtener la mejor solución posible en términos de función objetivo (Ecuación 1).

Variable: Xij: turno X asignado a la enfermera i el día j.

Parámetros:

- Información sobre instancia del problema:
    - E: Número de enfermeras
    - D: Número de días en el periodo de programación
    - T: Tipos de turno existentes

- H_T_F: Secuencias prohibidas de turnos

- Cantidad total mínima de turnos de cada tipo deseada y su correspondiente parámetro de flexibilidad:
  - V_min
  - b_min : Límite máximo faltantes aceptados. En % del promedio de turnos requeridos por día como mínimo.
- Cantidad total máxima de turnos de cada tipo deseada y su parámetro de flexibilidad.
  - V_max
  - b_max: Límite máximo sobrantes aceptados. En % del promedio de turnos requeridos por día como máximo.
- Matriz inicial auto-programada y parámetros de flexibilidad de preferencias:
  - X_i: Matriz inicial $E * D$.
  - V: Matriz lógica de turnos vetados $E * D$.
  - t_v_s: Número de turnos vetados por semana
  - a_p_g: Tolerancia para satisfacción de turnos.
  - b_p_g: Límite máximo aceptado para satisfacción de turnos.

Solución: La solución consiste en aplicar iterativamente la búsqueda ANS a la matriz X (inicialmente X_i) y obtener la mejor matriz X posible en términos de la función objetivo.

Resultados: Los resultados experimentales se pueden observar en el capítulo 7: Resultados experimentales, sección 7.1.

5.2.2 Implementación HABC. El método de solución HABC descrito en la sección 5.2 fue implementado en MATLAB (Ver anexo 2: Implementación HABC en MATLAB).Se utiliza la formulación del modelo expuesta en el capítulo 4. El objetivo, la variable y los parámetros de este problema son los mismos que en la sección anterior para el método ANS.

Solución: La única diferencia es claramente el método de solución cuyo algoritmo programado en MATLAB se muestra en el anexo 2.

Resultados: Los resultados experimentales de esta implementación se pueden ver en el capítulo 7: Resultados experimentales sección 7.3.

6 Resultados experimentales

6.1 Resultados ANS.

El modelo planteado fue resuelto por la técnica ANS (Sección 5.1) sobre tres instancias de la literatura (Gpost, Ortec y España, ver Anexo 3) . A continuación se muestran los resultados:

6.1.1 Problema inicial: asignar cantidad global de turnos. De acuerdo a la experimentación inicial se descubrió que es necesario modificar la búsqueda ANS complementándola previamente con otra búsqueda que resuelva el problema inicial de cantidad de turnos totales de cada tipo, descrito a continuación.

La programación de turnos de enfermería utilizando los movimientos planteados (para explorar el vecindario) en ambos métodos de solución generan un problema nuevo: El problema inicial de asignar la cantidad global de turnos de cada tipo, incluidos los turnos fuera. Esto debido a que los movimientos posibles solamente alternan turnos entre turnos **existentes** y no eliminan o crean ninguno nuevo. Por lo tanto la búsqueda de una solución para equilibrar los deseos de demanda y de satisfacción de preferencias estará sesgada hacia esta última si no se estudia inicialmente la cantidad de turnos de cada tipo (incluido el turno fuera).:

Es por lo tanto necesario el planteamiento del problema inicial de asignación de turnos totales:

Objetivo: El objetivo en este problema es satisfacer ambas restricciones de demanda y restricción de satisfacción de preferencias.

- Objetivo 1, Satisfacción de demanda: Se deben satisfacer lo máximo posible los requerimientos de demanda en términos de necesidades mínimas y máximas de personal por cada turno.
- Objetivo 2, Satisfacción de preferencias: Satisfacer lo máximo posible el horario inicial planteado por las enfermeras y sus requerimientos.

Variables:

- Cantidad total de turnos de cada tipo (incluyendo turnos fuera).

Parámetros iniciales:

- Cantidad total mínima de turnos de cada tipo deseada y su correspondiente parámetro de flexibilidad:

    - V_min
    - b_min : Límite máximo faltantes aceptados. En % del promedio de turnos requeridos por día como mínimo.
- Cantidad total máxima de turnos de cada tipo deseada y su parámetro de flexibilidad.
    - V_max
    - b_max: Límite máximo sobrantes aceptados. En % del promedio de turnos requeridos por día como máximo.

- Cantidad total máxima de turnos disponibles (depende del número total de enfermeras y de la intensidad horaria de cada una).
    - E
- Matriz inicial auto-programada y parámetros de flexibilidad de preferencias:
    - X_i: Matriz inicial $E * D$.
    - V: Matriz lógica de turnos vetados $E * D$.
    - a_p_g: Tolerancia para satisfacción de turnos.
    - b_p_g: Límite máximo aceptado para satisfacción de turnos.

Existe por lo tanto un problema multi-objetivo para definir el número total de turnos de cada tipo presentes en la matriz en todo momento. Los parámetros que expresan las ponderaciones sobre cada objetivo están expresados en términos de b_min y b_max para los deseos de demanda y en forma de a_p_g y b_p_g para los deseos de satisfacción de preferencias. Por lo tanto, siendo los demás parámetros con tendencia fija, el problema consiste en asignar la mayor cantidad de turnos de cada tipo a una matriz inicial, teniendo en cuenta las preferencias b_min y b_max de demanda y a_p_g y b_p_g de preferencias.

Solución

Se utiliza una búsqueda sencilla en el vecindario seleccionando en cada iteración el movimiento que tenga mayor valor en términos de función objetivo. El movimiento mencionado consiste en cambiar un turno de la matriz actual por un turno diferente y evaluar el puntaje de X con este cambio.

Resultados:

El modelo de solución anterior se aplicó al problema inicial de la instancia de prueba #1 y se obtuvo los siguientes resultados en X:

Tabla 2: Resultados de prueba, problema inicial

	X_i	X
miu_1	19,5	28
miu_2	19,5	28
miu_4	5	5
H_2_O_i	0	0
Total=	44	61

Tabla 3: Número de turnos inicial y final, problema inicial

	X_i	X
# Turnos F	28	28
# Turnos D	54	84
# Turnos N	58	28

Cómo se puede observar en las tablas 1 y 2, la solución al problema consistió en ajustar la cantidad de turnos D y N presentes en X a los valores exactamente requeridos (84 y 28). Los valores iniciales en X_i para D y N (54 y 58) presentaban sub-oferta y sobre-oferta de turnos, respectivamente. Los valores más de lo necesario para N, eran los valores que se necesitaban para D. Por lo tanto la búsqueda modifica la cantidad total de turnos presentes en la matriz para satisfacer lo máximo posible los requerimientos de demanda, de acuerdo a la función objetivo.

6.1.1.1 Experimentación problema inicial. El anterior problema inicial se implementa en MATLAB (Ver anexo 3: Implementación búsqueda inicial MATLAB) y se experimenta con dos técnicas de solución: ANS con problema inicial y ANS sin problema inicial para la instancia Gpost. A continuación se muestran los resultados en cada escenario:

Tabla 4: Instancia Gpost con solución problema inicial

	Valores iniciales	Valores de salida	Mejora	%Alcanzado del máximo - Eficacia
Valor función objetivo	40,9193	65,3697	60%	99%
sum_miu_1	19,5	28	44%	100%
sum_miu_2	19,5	28	44%	100%
sum_miu_3	3,3193	4,3697	32%	87%
sum_miu_4	5	5	0%	100%
c_pen_H1	0	0	0%	100%
c_pen_H2	6,4	0	-100%	100%

Tabla 5: Gpost sin Solución problema inicial:

	Valores iniciales	Valores de salida	Mejora	%Alcanzado del máximo - Eficacia
Valor función objetivo	40,9193	50,3697	23%	76%
sum_miu_1	19,5	20,5	5%	73%
sum_miu_2	19,5	20,5	5%	73%
sum_miu_3	3,3193	4,3697	32%	87%
sum_miu_4	5	5	0%	100%
c_pen_H1	0	0	0%	100%
c_pen_H2	6,4	0	-100%	100%

6.1.1.2 Análisis de resultados. Se puede apreciar que, para la instancia Gpost, la presencia de la búsqueda inicial altera significativamente la calidad de la mejor solución encontrada, al pasar de 50,3697 para la búsqueda ANS sin búsqueda inicial a 65,3697 ANS con búsqueda inicial. Esto se debe principalmente a que la búsqueda inicial permite movimientos de tipo eliminación y creación de nuevos turnos, por lo que los totales de cada turno cambian. Por el contrario, la búsqueda ANS no incorpora movimientos de este tipo en su proceso por lo que los totales de los turnos nunca cambiarán si solo se utilizara esta búsqueda.

En los experimentos siguientes de la sección 7.1 se incorpora la búsqueda inicial para que opere previamente a la búsqueda ANS.

6.1.2 **Resultados Gpost.** Esta instancia consiste en una cuadrilla de 6 enfermeras distribuidas en un periodo de programación de 28 días. Existen solo dos tipos de turno: día (D) y noche (N), no existe secuencia prohibida de los mismos, por lo que en esta instancia no aplica la restricción dura #1.

La demanda mínima y máxima es de 3 turnos tipo ‘D’ y 1 turno tipo ‘N’ para cada día de programación. Los datos se resumen en la tabla 1:

Tabla 6: Valores de demanda para Gpost

V_min, V_max :	**Día 1**	**Día 2**	**Día 3**	**Día …**	**Día 28**
D	3	3	3	…	3
N	1	1	1	…	1

En la ilustración 1 se muestra la matriz inicial auto-programada, el número total de turnos asignados en la matriz (incluyendo turnos fuera) es de 168 turnos los cuales están distribuidos como se muestra en la tabla 2.

Tabla 7: Distribución de turnos asignados en matriz inicial

Número de turnos asignados tipo 'D'	**57**
Número de turnos asignados tipo 'N'	61
Número de turnos asignados tipo 'F'	50
Número total de turnos asignados	168

6.1.2.1 Búsqueda inicial. Si se analizan los valores requeridos V_min (Tabla 1) para el periodo de programación, se necesita un total de 84 turnos 'D' y 28 turnos 'N' asignados en el periodo, por lo que los valores de 57 y 61 que tiene cada turno respectivamente en la matriz inicial, son insuficientes. Adicionalmente se puede saber la cantidad mínima de turnos fuera ('F') que se debe asignar en la matriz para satisfacer la restricción dura #2. Cada enfermera requiere como mínimo 4 turnos fuera programados en el periodo, uno para cada semana, para un total de 24 turnos fuera mínimos requerido. Por lo tanto se aprecia un exceso en la cantidad de turnos 'F' y 'N' y, por el contrario escasez en el número de turnos 'D' asignados en la matriz inicial. Se aplica la búsqueda inicial a la matriz, y se obtienen los siguientes valores:

Tabla 8: Número de turnos por tipo después de búsqueda inicial

Número de turnos asignados tipo 'D'	**84**
Número de turnos asignados tipo 'N'	28
Número de turnos asignados tipo 'F'	56
Número total de turnos asignados	168

La cantidad de turnos 'D' y 'N' fue modificada para satisfacer mejor los requerimientos de demanda. La cantidad restante de turnos es asignada a turnos tipos 'F': (ver tabla 10).

Número de turnos 'D'	**57**
Número de turnos 'N'	61
Número de turnos 'F'	50
Número total de turnos	168

Tabla 9 Cantidad Turnos Inicial

Número de turnos 'D'	**84**
Número de turnos 'N'	28
Número de turnos 'F'	56
Número total de turnos	168

Tabla 10: Cantidad turnos final

6.1.2.2 Búsqueda ANS. Una vez resuelto el problema inicial, se procede a mejorar el horario por medio de la técnica ANS descrita en la sección 4.1 (Búsqueda adaptativa). Se programaron un máximo de 1500 iteraciones con los siguientes parámetros de búsqueda:

Tabla 11: Parámetros de búsqueda ANS

# de iteraciones	**1500**
Betha_1	0.3
Betha_2	0.5

Juntando las dos búsquedas (Inicial y ANS) la función objetivo pasó de 40.9271 a 65.2514, produciendo el mayor puntaje posible en los valores de ajuste de las restricciones de demanda (28), los datos completos se pueden observar en la tabla 12.

Tabla 12: Resultados Gpost

	Inicial	**Final**
Restricción suave #1	19.5	28
Restricción suave #2	18	28
Restricción suave #3	0.6271	4.5714
Restricción suave #4	6	4.68
Restricción dura #2	3.2	0
Función objetivo	40.9271	65.2514

```
NNNDNDNNNFNDDNDDDNNNDDFNFNDF
DDFNDDDNNFFDDDNDDNDFDNNNNDFF
DFNDDNFFNNFNDFNDFDNDNNFFDFDF
DFDDNFDDDNNDNFDDDNNFFFDNDFDD
DNFFFNFFNFNFDNNNFFFFDFDNFNNN
DNNFDNDFNNNDFDFNFNNFFFFFDNDN
```

Figura 5: Matriz inicial Gpost

```
FNDDNDFNFFDDDNFFDNFNDNDDFDDF
FDFNDDDDDFFFDDDDNFDFNDNDNFFD
DFNDDDFDFDFNFFDDFDNDDDFFDDDF
DFDDFFNFDDDDNFDDDDDFFFDDDFND
NDFFFNDDDNNFDDFNDFFDDFDNFDFN
DDDFDFDFNDDDFDNFFDDDFDFFDNDD
```

Figura 6: Matriz final Gpost

6.1.2.3 Análisis. La instancia Gpost es una instancia pequeña de dificultad baja, la aplicación de la técnica de solución Búsqueda inicial + ANS arroja un resultado satisfactorio. Entre los datos por destacar se encuentran los óptimos alcanzados en las restricciones suaves 1 y 2, lo cual muestra una satisfacción completa de los requerimientos de demanda. También es destacable la mejora de la restricción relacionada con igualdad de carga laboral al pasar de 0.6 a 4,5 de 6 puntos posibles, lo cual muestra una mejora en la igualdad. Todos estos cambios fueron posibles solamente con una disminución de poco más de 1 en el valor de ajuste de la restricción suave #4 que mide satisfacción de preferencias. Se corrige la violación a la restricción dura #2 que estaba presente en la matriz inicial.

6.1.3 Resultados Ortec. La instancia Ortec consta de 13 enfermeras, para ser repartidas en 28 días de programación. Existen tres tipos de turno: ‘D’,’E’,’L’ y ‘N’. Las secuencias prohibidas de turno se resumen en la tabla 13:

Tabla 13: Secuencia de turnos prohibida Ortec

L-E
N-D
N-E
N-L

La demanda mínima y máxima deseada en el periodo es de:

- 1 turno ‘N’ todos los días.
- 3 turnos ‘D’, 3 turnos ‘E’ y 3 turnos ‘L’ todos los días, exceptuando los días del periodo múltiplos de 6 y 7, los cuales requieren solamente 2 turnos de cada uno (‘D’,’E’ y ‘L’).

En esta instancia se le da mayor importancia a satisfacer las restricciones de demanda (Restricciones suaves 1 y 2) y una menor importancia a las restricciones de igualdad y preferencias (Restricciones suaves 3 y 4). Los anteriores datos se introdujeron al algoritmo (Inicial + ANS) con los mismos parámetros de búsqueda mostrados en la tabla 3.

6.1.3.1 Búsqueda inicial.

De acuerdo a la información suministrada se necesita un total de 76 turnos de cada tipo ’D’,’E’ y ‘L’ y 28 turnos tipo ‘N’ asignados en la matriz, aplicando la búsqueda inicial los totales cambian:

Tabla 15: Número de turnos inicial

Número de turnos ‘D’	**73**
Número de turnos ‘E’	63
Número de turnos ‘L’	84
Número de turnos ‘N’	81
Número de turnos ‘F’	63
Número total de turnos	364

⇨

Tabla 14: Número de turnos final

Número de turnos ‘D’	**76**
Número de turnos ‘E’	76
Número de turnos ‘L’	76
Número de turnos ‘N’	28
Número de turnos ‘F’	108
Número total de turnos	364

6.1.3.2 Búsqueda ANS. Después de ambas búsquedas (Inicial y ANS) la función objetivo pasó de -10.8569 a 77.4146, los resultados se muestran en la tabla 15:

Tabla 16: Resultados Ortec

	Inicial	**Final**
Restricción suave #1	22.4219	27.7813
Restricción suave #2	17.5	27.7813
Restricción suave #3	6.6213	11.1250
Restricción suave #4	13	10.7271
Restricción dura #1	57.6	0
Restricción dura #2	12.8	0
Función objetivo	-10.8569	77.4146

La matriz de turnos final se muestra en la figura 5:

```
EELFDEFLDLFEENNFLLFDEFDEDLFF
NFEEDDFDEEFEFDLLLFLFNFDLNFDE
LLFLFLFLLDEEEFEFELDLLFFLFEEE
ENFLFEEENNFLFEDLFEFFFELFELDL
LLDEDEFDNFEDDFENNNNNFEFEFFEF
FENFFFDDLFDDFEFEEDEFDLNFLDLL
DLDDNFFEDLNFDDLFFLFEFDLDENNF
LDLFEDEFEFELLFLLFEDDFDEDLFFF
DFLDLFDEDLDDFLFEDDLFLDEFEDFF
EDEDLLFLFDDFLLDEDFDEFNFDLLFF
FFDLLFLFLDLLFFDDEEELFLDLDDFD
FEFFENNNFELFFFEDDDLFDEFEDELD
DDEEEFLFEELNNFFDLFEFELLNFEFN
```

Figura 5: Matriz final Ortec

6.1.3.3 Análisis. La dificultad aumenta en la instancia Ortec respecto a la anterior. El número de enfermeras es mayor, así como el número de tipos de turno. La restricción dura #1 opera en esta instancia debido a las secuencias prohibidas de turnos que contiene. De nuevo las restricciones de demanda son satisfechas hasta valores cercanos a su máximo posible, la igualdad se duplica al pasar de 6.6213 a 11,1250 de 13 posibles y la restricción de preferencias disminuye solamente en menos de 3 para tomar el valor de 10.7271. Algo importante a resaltar es que la búsqueda elimina las altas violaciones a las restricciones duras 1 y 2 presentes en la matriz inicial.

6.1.4 Resultados España. La instancia España consiste en asignar 16 enfermeras en un periodo de programación de 31 días. Los tipos de turno son: ‘M’, ‘A’, ‘N’, ‘D’ y ‘H’. Las restricciones de secuencia de turnos se muestran en la siguiente tabla:

Tabla 17: Secuencia prohibida de turnos instancia España

N-M
N-A
N-D
N-H
D-M
D-A
D-D
D-H
H-M
H-A
H-D
H-H

- La demanda requerida por día en el periodo de programación es de 4 turnos tipo ‘M’, 4 turnos tipo ‘A’, 2 turnos tipo ‘N’, 1 turno tipo ‘D’ y 4 turnos tipo ‘H’ solamente en los días múltiplos de 5 y 6 del periodo de programación.
- Esta instancia le da mayor importancia a cumplir la restricción suave #1, en orden de importancia le siguen las restricciones 2 y 3 y por último, la de menor importancia en este caso, es la restricción #4. En otras palabras evitar la sub-programación tiene una prioridad de 1, evitar la sobre-programación y aumentar la igualdad tienen ambas prioridad de 2, y por último satisfacer las preferencias de los empleados tiene una prioridad de 3.

6.1.4.1 Búsqueda inicial. De acuerdo a la información suministrada se necesitan 124 turnos asignados tipo ‘M’, 124 tipo ‘A’, 62 tipo ‘N’, 31 tipo ‘D’ y 36 tipo ‘H’. Se aplica la búsqueda inicial y se obtienen nuevos totales de turnos, se muestran en la tabla 18.

Tabla 19: Número de turnos inicial

Número de turnos ‘M’	**94**
Número de turnos ‘A’	86
Número de turnos ‘N’	77
Número de turnos ‘D’	86
Número de turnos ‘H’	90
Número de turnos ‘F’	63
Número total de turnos	496

⇨

Tabla 18: Número de turnos final

Número de turnos ‘M’	**121**
Número de turnos ‘A’	121
Número de turnos ‘N’	60
Número de turnos ‘D’	31
Número de turnos ‘H’	32
Número de turnos ‘F’	131
Número total de turnos	496

6.1.4.2 Búsqueda ANS. Aplicando la búsqueda inicial + ANS se obtuvo resultados mostrados en la tabla 20:

Tabla 20: Resultados instancia España

	Inicial	**Final**
Restricción suave #1	21.7082	29.6021
Restricción suave #2	17.1034	30.5889
Restricción suave #3	6.5325	15.3799
Restricción suave #4	16	9.12
Restricción dura #1	126	0
Restricción dura #2	26	0
Función objetivo	-90.6559	84.6909

La matriz de turnos final se muestra en la figura 6:

```
MDDNFMFAFMFAMMHFMAMADFFANNFMAMA
ANFMHHFMMMAMFAHNNNFMAAAFMMMFAMA
MAFAMAMAAFFHFAHFMMMMMDFAAHFMAAM
DFMMAMFMMMFMFMDNFMAAAMMFMMAMHFD
AAFAAAAMFDFADFAMDFMNFAMMMHFAMNF
NNNFHNNNFNNNNFMMMAAFMMAMDFMAFAA
MFAAAMMFDFAMHNNFFAMAAAFDFHNFMAM
AMMFMAMNNFMMADNFAADFMFDNFMAAMMA
MAAFMAAMFNNFMFMAAFHHNNFFAHHFNFM
AAFAAMADFAADFMAAMFNNFMMAFAHFDNF
FHFMNFAHNFMAMMMAFMAHFMMANFAAFDN
FFMFMDDFMFAHFAADFFAMAFAFFDDDFFF
NFAMHNNFAAFAHFMMFDNFMNFMAAHFMAF
FMAFNFMMMAMHFNFANFHHFFAMAANNFMD
FMNNFHFAAADFAFHFANFANFNNFAHNNFM
FMMDDFFAAMMFNFAMAMHHFANFMMAMAFN
```

Figura 6: Matriz final España

6.1.4.3 Análisis. La instancia España contiene 16 enfermeras y 5 tipos de turno, 12 secuencias restringidas de turno, lo que la convierte en una instancia con una dificultad mayor. De nuevos las restricciones de demanda son altamente satisfechas y la igualdad en la programación aumenta considerablemente. Cabe resaltar que la restricción número disminuyó en una mayor proporción que las demás instancias, al pasar de 16 a 9,12. Esto se explica por la característica restringida en la secuencia de turnos en esta instancia, combinado con la prioridad número 4 (última) para ser satisfecha con respecto a las demás. La búsqueda inicial + ANS logra mejorar la función objetivo notablemente y corregir al mismo tiempo las violaciones efectuadas a las restricciones duras.

6.2 Resultados HABC

Al implementar la técnica de solución HABC el recurso computacional aumentó considerablemente, esto debido a varios motivos:

- La técnica de solución es poblacional, por lo que, a diferencia de ANS, HABC mejora iterativamente diez soluciones al tiempo. Considerando esto solamente, el tiempo de cálculo de una iteración aumenta diez veces más.
- El algoritmo HCO utilizado en la etapa obrera de HABC es más exigente computacionalmente que la búsqueda obrera tradicional en el ABC ya que mejora cada solución a su máximo local. Esto combinado con la característica poblacional del algoritmo hacen que el tiempo de cómputo para cada iteración aumente considerablemente con respecto al ANS.

Tomando en cuenta estas consideraciones se evaluaron las instancias Gpost y Ortec aplicando el HABC con 50 iteraciones como límite de parada.

6.2.1 Resultados Gpost. La función objetivo pasó de 40.9271 a 64.1841, produciendo el mayor puntaje posible en el valor de ajuste de la restricción de demanda #1 (sub-programación), los datos completos se pueden observar en la tabla 3. Se corrieron 50 iteraciones del algoritmo.

Tabla 21: Resultados HABC instancia Gpost

Tabla 3: Resultados Gpost	**Inicial**	**Final**
Restricción suave #1	19.5	28
Restricción suave #2	18	26.5
Restricción suave #3	0.6271	4.6441
Restricción suave #4	6	5.040
Restricción dura #2	3.2	0
Función objetivo	40.9271	64.1841

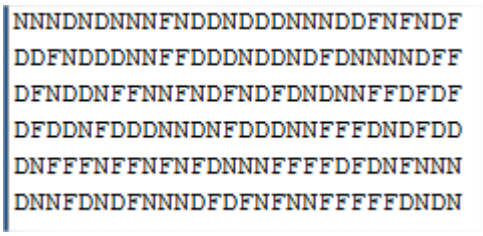


Figura 7: Matriz inicial Gpost

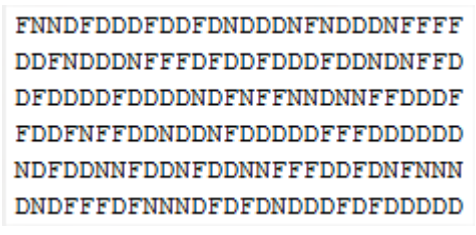


Figura 8: Matriz final Gpost

6.2.2 Resultados Ortec. Después de aplicar la búsqueda HABC la función objetivo pasó de -10.8569 a 69.6626. Se corrieron 50 iteraciones del algoritmo. En la siguiente tabla se muestran los resultados por restricción:

Tabla 22: Resultados HABC instancia Ortec

Tabla 3: Resultados Ortec	**Inicial**	**Final**
Restricción suave #1	22.4219	26.6875
Restricción suave #2	17.5	27.7656
Restricción suave #3	6.6213	11.5382
Restricción suave #4	13	9.6713
Restricción dura #1	57.6	0
Restricción dura #2	12.8	0
Función objetivo	-10.8569	69.6626

La matriz de turnos final se muestra en la figura 9:

```
EELDDEFLLLNNFNFELFEDELDFDLDE
NFEEEENFELFEEELLLFLLNNFLNFDE
FLDLLLNFDEENNNFEELNFLLLLNFDE
LNNNFEEENNFLNNNNNFENFNNFELLL
LNNNFELDEDNFDFEENNNFDEEEFEEN
LLNNFEDDLFDDEEFLNNFDDENFEDLL
DEEENNFEDFEEDLLNFLDEFDLDNNNF
LDLFEEELFEELLFLLDENFLFDELLLN
LLDDDFDEDNFDLLFLDDLNFDLDLDLD
EDDDLDFDNFDELDEFELDEENFNFLLN
EDLLLFFLLDLNNFEDEEEEFLDLDDDD
DEFLDLDNFELDNFDDDDLNFDEDDELD
DDENFDLFELLLNNFDLDDLFENNFEEN
```

Figura 9: Matriz final Ortec

7 Conclusiones

- De acuerdo a la revisión de la literatura efectuada sobre el NSP se encontró que un problema común entre los investigadores es la adaptación de las técnicas de solución a las distintas formulaciones del modelo, debido a la gran variedad de restricciones presentes entre un problema y otro. Si bien se evidenciaron esfuerzos en la literatura para crear formulaciones de modelo genéricas, estas no son suficientes para abarcar la totalidad de variedad en términos de restricciones especialmente de preferencias de empleados.
- De acuerdo a la revisión de literatura se encontró que la robustez de una técnica de solución para NSP de ve disminuida debido a la incorporación de restricciones específicas que limitan o dificultan la aplicación de la técnica en otros modelos diferentes a los planteados.
- La solución del NSP utilizando auto-programación ha sido estudiada recientemente en la literatura en entornos reales como método de solución, cuya solución inicial es la matriz auto-programada.
- De acuerdo a la revisión de literatura sobre NSP se observó una tendencia creciente por aumentar y mejorar la representación y satisfacción de las preferencias de los empleados en el modelo.
- El modelo planteado en este trabajo fue programado y probado en MATLAB con una serie de instancias de la literatura que variaban entre sí en términos de tamaño y complejidad.

- El método demostró ser posible de ser implementado en un marco de solución de técnicas modernas al problema NSP (ANS, HABC) cuya creación inicial fue basada en modelos con métodos tradicionales de enumeración de restricciones.
- Comparado con los modelos tradicionales, la utilización del método de auto-programación y la utilización de solo dos tipos de funciones de ajuste para representar la calidad del horario en cualquier momento permitieron utilizar el modelo para evaluar una variedad de instancias en tamaño y complejidad, sin necesidad de programar restricciones adicionales de acuerdo a cada problema.
- Para las instancias evaluadas, el método de búsqueda adaptativa en el vecindario (ANS) resulta en una mayor eficiencia computacional que la búsqueda poblacional híbrida HABC.
- La utilización del método de solución del NSP planteado en este trabajo (Formulación y técnica de solución) permitió mejorar considerablemente las soluciones inicialmente planteadas por las enfermeras en múltiples instancias, en términos de satisfacción de demanda y preferencias.
- En los modelos tradicionales las restricciones de satisfacción de empleados son enumeradas por el programador o investigador, por lo que estas restricciones pueden no reflejar las preferencias de todas las enfermeras. Utilizando el modelo de formulación del problema expuesto en este trabajo se logra aumentar la representación de las preferencias reales de cada enfermera en cada periodo de programación. El resultado es un modelo más sencillo en términos de cantidad de restricciones y una representación más cercana a las preferencias reales de las enfermeras en cada periodo.

- Con la realización de este trabajo se muestra que es posible resolver un problema de NSP con una mayor inclusión de los empleados y a la vez una mayor representación de sus preferencias en los modelos de solución, sin alterar las necesidades de demanda.
- El modelo completo aquí desarrollado puede ser utilizado fácilmente, reemplazando a la técnica manual de programación, en múltiples escenarios con NSP debido a su flexibilidad y robustez.

8 Recomendaciones

- Al utilizar una formulación del modelo basada en auto-programación y aplicarla a una técnica de solución específica, se debe tener en cuenta el problema inicial de cantidad total de turnos descrito en la sección XX.
- El componente práctico de este trabajo puede ser comprobado en un entorno laboral real de NSP y comprobar su efectividad.

Referencias bibliográficas

Ásgeirsson, E. I. (2014) bridging the gap between self schedules and feasible schedules in staff scheduling. *annals of operations research*, *218(1)*, 51-69. http://doi.org/10.1007/s10479-012-1060-2

Asta, S., Özcan, E., y Curtois, T. A (2016) tensor based hyper-heuristic for nurse rostering. *knowledge-based systems, 98*, 185–199. http://doi.org/10.1016/j.knosys.2016.01.031

Awadallah, M. A.; Bolaji, A. L. A. y Al-betar, M. A. (2015) A hybrid artificial bee colony for a nurse rostering problem. *Applied Soft Computing Journal*, *35*, 726–739. http://doi.org/10.1016/j.asoc.2015.07.004

Awadallah, A., Khader, A., Al-betar y M., Bolaji, A. (2014) Harmony search with novel selection methods in memory consideration for nusr rostering problem. *Asia-Pacific Journal of Operational Research, 31*(3). DOI: 10.1142/S0217595914500146

Ayob, M.; Hadwan, M.;Nazri N. Z. A. y Ahmad, Z. (2013) Enhanced harmony search algorithm for nurse rostering problems. *Journal of Applied Sciences*, *1*(6), p846–853. http://doi.org/10.3923/jas.2013.846.853

Baeklund, J. (2013) Nurse rostering at a Danish ward. *Annals of Operations Research*, *Dec*, 1–17. http://doi.org/10.1007/s10479-013-1511-4

Bai, R., Burke, EK., Kendall, G., Li, J. y McCollum, B. (2010) A hybrid evolutionary approach to the nurse rostering problem. *IEEE Transactions on Evolutionary Computation*, *14*(4) 580–590. http://doi.org/10.1109/TEVC.2009.2033583

Bautista, Karina. (2014) *Estado del arte de la logística hospitalaria* (tesis de pregrado). Universidad Industrial de Santander, Bucaramanga. Disponible en el catálogo en línea de la biblioteca de la Universidad Industrial de Santander: http://tangara.uis.edu.co/.

Bilgin, B., De Causmaecker, P., Rossie, B. y Vanden Berghe, G. (2010) Local search neighbourhoods for dealing with a novel nurse rostering model. *Ann Oper Res. (194)*, 33–57. http://doi.org/DOI 10.1007/s10479-010-0804-0

Bolaji, A.L., Khader, A.T., Al-betar, M.A. y Awadallah, A. (2014) University course timetabling using hybridized artificial bee colony with hill climbing optimizer. *Journal of Computational Science, 5*(5), 809-818. https://doi.org/10.1016/j.jocs.2014.04.002

Brucker, P., Burke, EK., Curtois, T., Qu, R., y Vanden Berghe, G. (2010). A shift sequence based approach for nurse scheduling and a new benchmark dataset. *Journal of Heuristics*, *16*(4), 559–573. http://doi.org/10.1007/s10732-008-9099-6

Burke, E. K., y Curtois, T. (2014). New approaches to nurse rostering benchmark instances. *European Journal of Operational Research*, *237*(1), 71–81. http://doi.org/10.1016/j.ejor.2014.01.039

Burke, E. K., Curtiois, T., Qu, R. y Vanden Berghe G. (2007). A Scatter Search Approach to the Nurse Rostering Problem, 1–25.

Burke, E. K.; Li, J., y Qu, R. (2010). A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems. *European Journal of Operational Research. 203*(2) 484–493. http://doi.org/10.1016/j.ejor.2009.07.036

Burke, E. K.; Li, J. y Qu, R. (2012) A Pareto-based search methodology for multi-objective nurse scheduling. *Annals of Operations Research*, *196*(1) 91–109. http://doi.org/10.1007/s10479-009-0590-8

Burke, E. K., Curtois, T., van Draat, LF, Ommeren, J-K. y Post, G. (2011) Progress control in iterated local search for nurse rostering. *The Journal of the Operational Research Society, 62*(2), 360–367. http://doi.org/10.1057/jors.2010.86

Chiaramonte, M., Cochran, D.,. y Caswell, D. (2014). Nurse preference rostering using agents and iterated local search. *Annals of Operations Research, 226*(1), p 443–461. http://doi.org/10.1007/s10479-014-1701-8

Constantino, A. A., Landa-Silva, D., de Melo, E.L., de Mendonca., C., Rizzato, D. y Romao, W. (2014) *A heuristic algorithm based on multi-assignment procedures for nurse scheduling. Annals of Operations Research. 218*(1), 165–183. http://doi.org/10.1007/s10479-013-1357-9

De causmaecker, P., y Vanden Berghe, G. (2011) *A categorisation of nurse rostering problems. En: Journal of Scheduling. 14*(1), 3–16. http://doi.org/10.1007/s10951-010-0211-z

Della Croce, F., y Salassa, F. (2014). *A variable neighborhood search based matheuristic for nurse rostering problems. Annals of Operations Research, 21*(81), 185–199. http://doi.org/10.1007/s10479-012-1235-x

Díaz, Johanna y Pinto, Karen. (2009). *Diseño y validación de un modelo matemático para el proceso de asignación de salones en el edificio Camilo Torres de la Universidad Industrial De Santander* (tesis de pregrado). Universidad Industrial de Santander, Bucaramanga. Disponible en el catálogo en línea de la biblioteca de la Universidad Industrial de Santander: http://tangara.uis.edu.co/.

Dorne, R. (2008). *Personnel Shift Scheduling and Rostering. Service Chain Management.* 125-138. Springer.. http://doi.org/10.1007/978-3-540-75504-3. ISSN: 08848289.

Drake, R. G. (2014). The "Robust" roster: Exploring the nurse rostering process. *Journal of Advanced Nursin. 70*(9), 2095–2106. http://doi.org/10.1111/jan.12367

Drake, R. G. (2014). The nurse rostering problem: From operational research to organizational reality? *Journal of Advanced Nursing*, *70*(4), 800–810. http://doi.org/10.1111/jan.12238

Glass, C. A., y Knight, R. A. (2010). The nurse rostering problem: A critical appraisal of the problem structure. *European Journal of Operational Research*, *202*(2), 379–389. http://doi.org/10.1016/j.ejor.2009.05.046

Haspeslagh, S.;De causmaecker, P.; Schaerf, A.y Stølevik, (2014) M. The first international nurse rostering competition 2010. *Annals of Operations Research. 218*(1), 221–236. http://doi.org/10.1007/s10479-012-1062-0

He, F., y Qu, R. (2012) A constraint programming based column generation approach to nurse rostering problems. *Computers and Operations Research*, *39*(12), 3331–3343. http://doi.org/10.1016/j.cor.2012.04.018

Hillier, Frederick S. y Lieberman, Gerald J. (2010) *Introducción a la investigación de operaciones.* 9a edición. México: Mc Graw Hill Companies. ISBN 978-607-15-0308-4

Huang, H. et al. (2014) An evolutionary algorithm based on constraint set partitioning for nurse rostering problems. *Neural Computing and Applications*, 1–13. http://doi.org/10.1007/s00521-013-1536-2

Idalia, Flor. (2015) *Programación dinámica.* Universidad Autónoma de México. http://www.ingenieria.unam.mx/sistemas/PDF/Avisos/Seminarios/SeminarioV/Sesion6_IdaliaFlores_20abr15.pdf.

Ismail, W. R., y Jenal, R. (2013) Master plan nurse duty roster using the 0-1 goal programming technique. *AIP Conference Proceedings*, *1522*, 1394–1400. http://doi.org/10.1063/1.4801292

Legrain, A.; Bouarab, H. y Lahrichi, N. (2015). The nurse scheduling problem in real-life. *Journal of Medical Systems, 39*(1), p 160. http://doi.org/10.1007/s10916-014-0160-8

Leksakul, K. y Phetsawat, S. (2014) Nurse scheduling using genetic algorithm. *Mathematical Problems in Engineering*, http://doi.org/10.1155/2014/246543

Liang, B. y Turkcan, (2015). An Acuity-based nurse assignment and patient scheduling in oncology clinics. *Health Care Management Science*. http://doi.org/10.1007/s10729-014-9313-z

Lin, C-C., Kang, J-R, Liu, W-Y. Deng, D-J. (2014). Modelling a Nurse Shift Schedule with Multiple Preference Ranks for Shifts and Days Off. *Mathematical Problems in Engineering, 2014.* http://dx.doi.org/10.1155/2014/937842

Lü, Z., y Hao, J. K. (2012) Adaptive neighborhood search for nurse rostering. *European Journal of Operational Research*. 218(3), 865–876. http://doi.org/10.1016/j.ejor.2011.12.016

M'Halla H, R. y Alkhabbaz, A. (2013). Scheduling of nurses: A case study of a Kuwaiti health care unit. *Operations Research for Health Care. 2*(1-2), 1–19. http://doi.org/10.1016/j.orhc.2013.03.003

Maenhout, B. y Vanhoucke, M. (2013). An integrated nurse staffing and scheduling analysis for longer-term nursing staff allocation problems. *Omega (United Kingdom), 41*(2), 485–499. http://doi.org/10.1016/j.omega.2012.01.002

Maenhout, B. y Vanhoucke, M. (2010) Branching strategies in a branch-and-price approach for a multiple objective nurse scheduling problem. *Journal of Scheduling, 13*(1), 77–93. http://doi.org/10.1007/s10951-009-0108-x

Martin, S., Ouelhadj D., Smet, P., Vanden Berghe G. y Özcan, E. (2013). Cooperative search for fair nurse rosters. *Expert Systems with Applications. 40*(16), 6674–6683. http://doi.org/10.1016/j.eswa.2013.06.019

Merino Maestre, María. (2016) *Técnicas clásicas de optimización. Parte I programación Lineal y no lineal.* Recuperado de*:* http://www.ehu.eus/mae/html/prof/Maria_archivos/plnlapuntes.pdf

Mutingi, M., & Mbohwa, C. (2015). A multi-criteria approach for nurse scheduling fuzzy simulated metamorphosis algorithm approach. *Mutingi, M., & Mbohwa, C , IEOM 2015 - 5th International Conference on Industrial Engineering and Operations Management* http://doi.org/10.1109/IEOM.2015.7093904

Petrovic, S. y Vanden Berghe, G. (2012) A comparison of two approaches to nurse rostering problems. *Annals of Operations Research, 194*(1), 365–384. http://doi.org/10.1007/s10479-010-0808-9

Rais, A. y Viana, A. (2011) Operations research in healthcare: A survey. *International Transactions in Operational Research, 18*(1), 1–31. http://doi.org/10.1111/j.1475-3995.2010.00767.x

Ramsey-Coleman, Joyce. (2012) Staff scheduling synchronicity. *Health Management Technology*, 346

Rönnberg, E. y Larsson, T. (2010). Automating the self-scheduling process of nurses in Swedish healthcare: A pilot study. *Health Care Management Science. 13*(1), 35–53. http://doi.org/10.1007/s10729-009-9107-x

Smet, P., Bilgin, B., De Causmacker, P. y Vanden Berghe G. et al. Modelling and evaluation issues in nurse rostering. *Annals of Operations Research*, *218*(1), 303–326. http://doi.org/10.1007/s10479-012-1116-3

Tassopoulos, I. X.; Solos, I. P. y Beligiannis, G. N. (2015) A two-phase adaptive variable neighborhood approach for nurse rostering. *Computers and Operations Research.* http://doi.org/10.1016/j.cor.2015.02.009

Todorovic, Nikola y Petrovic, Sanja. (2013). Bee Colony Optimization Algorithm for Nurse Roster. *IEE transactions on systems, man, and cybernetics: SYSTEMS, 43*(2), 467–73.

Tontarski, Clayton. (2015) *Modeling and Analysis of OR Nurse Scheduling Using Mathematical Programming and Simulation.* (Tesis de maestría). 2015. Universidad de ingeniería Kate Gleason. Departamento de ingeniería industrial y de sistemas. Rochester. Estados Unidos de América.

Topaloglu, S. y Selim, H. (2010). Nurse scheduling using fuzzy modeling approach. *Fuzzy Sets and Systems*, *161*(11), 1543–1563. http://doi.org/10.1016/j.fss.2009.10.003

Tsai, C. y Lee, C. (2010). Optimization of Nurse Scheduling Problem with a Two-Stage Mathematical Programming Model. *Asia Pacific Review*, *15*(4), 503–516.

Universidad de Granada (2016). *Branch & Bound.* http://elvex.ugr.es/decsai/algorithms/slides/5%20Branch%20and%20Bound.pdf.

Universidad de Granada (2016). *Programación dinámica. Análisis y diseño de algoritmos.* http://elvex.ugr.es/decsai/algorithms/slides/6%20Dynamic%20Programming.pdf

Valouxis, C., Gogos, C., Goulas, G., Alefragis, P. y Housos, E. (2012). A systematic two phase approach for the nurse rostering problem. *European Journal of Operational Research. 219*(2), 425–433. http://doi.org/10.1016/j.ejor.2011.12.042

Wong, T. C., Xu, M. y Chin, K.S. (2014). A two-stage heuristic approach for nurse scheduling problem: A case study in an emergency department. *Computers and Operations Research*, *51*, p 99–110. http://doi.org/10.1016/j.cor.2014.05.018

Wright, P. D. y Mahar, S. (2013). Centralized nurse scheduling to simultaneously improve schedule cost and nurse satisfaction. *Omega (United Kingdom), 41*(6), 1042–1052. http://doi.org/10.1016/j.omega.2012.08.004

Wu, T. H., Yeh, J. Y. y Lee, Y. M. (2015). A particle swarm optimization approach with refinement procedure for nurse rostering problem. *Computers and Operations Research*, *54*, 52–63. http://doi.org/10.1016/j.cor.2014.08.016

Yilmaz, E. (2012). A mathematical programming model for scheduling of nurses' labor shifts. *Journal of Medical Systems, 36*(2) 491–496. http://doi.org/10.1007/s10916-010-9494-z

Zhou, J. F., Fan, Y. Y. y Zeng, H. Z. (2012). A Nurse Scheduling Approach Based on Set Pair Analysis. *International Journal of Industrial Engineering-Theory Applications and Practice, 19*(9), 359–368.

Apéndices

8.1 Apéndice A: Algoritmo ANS en MATLAB:

```
function[X]=busq_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,
X_ult_7_dias,T,H_T_F,a_p_g,b_p_g,c_parada_ANS,inst)
global X_tabu X_re
X_mejor_mejor=X;
fo_mejor_mejor=-1000;
R=100;%Número de veces seguidas en las que no hay mejora.
c_parada_r=100;%Criterio de parada en número de reinicios.
reinicios=0;
fo_mejor=zeros(1,c_parada_ANS);
%Inicialización de variables.
it=0;
dl=0;
contador=0;
contador_r=0;
theta=round((size(X,1)*3)/10);
betha_1=0.3;
betha_2=0.5;
%ANS
while                                  it<c_parada_ANS&&reinicios<c_parada_r
fo_inicial=funcion_obj_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,
V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
    if dl<betha_1
        if ~strcmp(m_dl,'inten')
        X_tabu=zeros(size(X));
        X_tabu(:,:,1:size(T,1)+1)=0;
        end

      X=inten(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult_7
      _dias,T,H_T_F,a_p_g,b_p_g);
        m_dl='inten';
    elseif dl>=betha_1&&dl<betha_2
        if ~strcmp(m_dl,'interme')
        X_re=zeros(size(X));
        X_re(:,:,1:size(T,1)+1)=0;
        end

      X=interme(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult
      _7_dias,T,H_T_F,a_p_g,b_p_g);
        m_dl='interme';
    else

      X=diversi(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult
      _7_dias,T,H_T_F,a_p_g,b_p_g);
        m_dl='diversi';
```

```
    end
    it=it+1;
[fo_final,miu_1,miu_2,miu_3,miu_4,c_pen_H1,c_pen_H2]=funcion_obj_ANS(X,X_
i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_
g,b_p_g);
    if fo_final<=fo_inicial
        contador=contador+1;
        if contador==theta
            dl=dl+(1-dl)/6;
            contador=0;
        end
    else
        dl=dl-dl/10;
        contador=0;
    end
    if fo_final<=fo_inicial
        contador_r=contador_r+1;
    else
        contador_r=0;
    end
    if contador_r==R
        dl=1;
        reinicios=reinicios+1;
        if rand>0.5
        X=X_mejor_mejor;
        end
        contador_r=0;
    end
    if fo_final>fo_mejor_mejor
    X_mejor_mejor=X;
  [fo_mejor_mejor]=funcion_obj_ANS(X_mejor_mejor,X_i,
V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,
b_p_g);
    end
    o_dl(it)=dl;
    fo_mejor(it)=funcion_obj_ANS(X,X_i,
V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,
b_p_g);
end

X=X_mejor_mejor;
```

8.2 Apéndice B: Algoritmo HABC en MATLAB

```
function
[]=HABC(X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult_7_dias,T,
H_T_F,a_p_g,b_p_g,el_inst)
SN=10;%Número de soluciones presentes en FSM.
MCN=50;%Número máximo de iteraciones.
limit=100;%Iteraciones sin mejora.
HCR=0.5;%Parámetro de búsqueda de HCO.
%Generar FSN
[FSM]=gen_FSM(X_i,V_min,T,SN);
f_obj_r_FSM=zeros(1,SN);
for                                                                    i=1:SN
f_obj_r_FSM(i)=funcion_obj_ANS(FSM(:,:,i),X_i,V_min,a_min,b_min,a_max,b_m
ax,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
end
[f_obj_r_FSM,I]=sort(f_obj_r_FSM,'descend');%f_obj_r_FSM son los valores
f_obj ordenados en orden descendente.
FSM=FSM(:,:,I);
K=1:SN;
trials=zeros(size(K));
f_obj_ant=f_obj_r_FSM;
K_ant=K;
for it=1:MCN
    %HCO
    phi=0.4;
    dens=(sum(sum(V_min)))/(size(FSM,1)*size(FSM,2));
    q=1-phi*dens;
    for k=1:SN
        if rand<HCR
            X=FSM(:,:,k);
            X_mejor=X;
            c=0;
            while c==0
                [X]=mov(X)
f_actual=funcion_obj_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,
a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);

f_mejor=funcion_obj_ANS(X_mejor,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a
_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
                if f_actual<=f_mejor
                    c=1;
                    FSM(:,:,k)=X_mejor;
                else
                    X_mejor=X;
                end
            end
            if
funcion_obj_ANS(FSM(:,:,k),X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,
a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g)<=f_obj_ant(find(K_ant==K(k)))
                trials(K(k))=trials(K(k))+1;
            else
```

```
                trials(K(k))=0;
            end
        end
    end
%%Ordenar FSM
for                                                                 i=1:SN
f_obj_r_FSM(i)=funcion_obj_ANS(FSM(:,:,i),X_i,V_min,a_min,b_min,a_max,b_m
ax,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
end
[f_obj_r_FSM,I]=sort(f_obj_r_FSM,'descend');%f_obj_r_FSM son los valores
f_obj ordenados en orden descendente.
FSM=FSM(:,:,I);
K=K(I);

f_obj_ant=f_obj_r_FSM;
K_ant=K;

%Termina HCO

%Inicia observadora
p=f_obj_r_FSM./sum(f_obj_r_FSM);
sum_prob=0;
o=10;

r=rand;
c=0;
for i=1:SN
    sum_prob=sum_prob+p(i);
    if 1-sum_prob<=r
        c=1;
        o=i;
    end
    if c==1
        break
    end
end
X=FSM(:,:,o);
X_inicial=X;
[X]=mov(X);
if
funcion_obj_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult
_7_dias,T,H_T_F,a_p_g,b_p_g)>funcion_obj_ANS(X_inicial,X_i,V_min,a_min,b_
min,a_max,b_max,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
   FSM(:,:,o)=X;
end
if
funcion_obj_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult
_7_dias,T,H_T_F,a_p_g,b_p_g)<=f_obj_ant(find(K_ant==K(o)))
   trials(K(o))=trials(K(o))+1;
else
    trials(K(o))=0;
end

%%Ordenar FSM
```

```
for                                                                    i=1:SN
f_obj_r_FSM(i)=funcion_obj_ANS(FSM(:,:,i),X_i,V_min,a_min,b_min,a_max,b_m
ax,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
end
[f_obj_r_FSM,I]=sort(f_obj_r_FSM,'descend');%f_obj_r_FSM son los valores
f_obj ordenados en orden descendente.
FSM=FSM(:,:,I);
K=K(I);
[fo_mejor_mejor,P]=max(f_obj_r_FSM);
X_mejor_mejor=FSM(:,:,P);
o_fo_mejor_mejor(it)=fo_mejor_mejor;
%Termina onlooker

%Inicia exploradora
s=randi(1,SN);
if trials(s)>=limit
    X=FSM(:,:,find(K==s));
    %% Realiza movimiento aleatorio
    if rand<q
        ran2=rand;
        if ran2<0.4
            r=randi(1,size(X,2));
            EE_1=find(X(:,r)~='F');
            EE_2=find(X(:,r)=='F');
            if length(EE_1)>=1&&length(EE_2)>=1
                e=randi(1,length(EE_1));
                e2=randi(1,length(EE_2));
                X=X_inicial;
                aux=X(EE_1(e),r);
                X(EE_1(e),r)=X(EE_2(e2),r);
                X(EE_2(e2),r)=aux;
            end

        else
            e=randi(1,size(X,1));
            DD_1=find(X(e,:)~='F');
            DD_2=find(X(e,:)=='F');
            if length(DD_1)>=1&&length(DD_2)>=1
                d=randi(1,length(DD_1));
                d2=randi(1,length(DD_2));
                X=X_inicial;
                aux=X(e,DD_1(d));
                X(e,DD_1(d))=X(e,DD_2(d2));
                X(e,DD_2(d2))=aux;

            end
        end
    else
        ran3=rand;
        if ran3<0.4
            d=randi(1,size(X,2));
            EE=find(X(:,d)~='F');
            if length(EE)>1
                perm_EE=nchoosek(EE',2);
            else
                perm_EE=[];
```

```
            end
            i=randi(1,size(perm_EE,1));
            X=X_inicial;
            aux=X(perm_EE(i,1),d);
            X(perm_EE(i,1),d)=X(perm_EE(i,2),d);
            X(perm_EE(i,2),d)=aux;
        else
            e=randi(1,size(X,1));
            DD=find(X(e,:)~='F');
            if length(DD)>1
                perm_DD=nchoosek(DD,2);
            else
                perm_DD=[];
            end
            i=randi(1:size(perm_DD,1));
            X=X_inicial;
            aux=X(e,perm_DD(i,1));
            X(e,perm_DD(i,1))=X(e,perm_DD(i,2));
            X(e,perm_DD(i,2))=aux;
        end
    end
    FSM(:,:,find(K==s))=X;
    trials(s)=0;
    if
funcion_obj_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,a_i,V,a_p,X_ult
_7_dias,T,H_T_F,a_p_g,b_p_g)>fo_mejor_mejor

fo_mejor_mejor=funcion_obj_ANS(X,X_i,V_min,a_min,b_min,a_max,b_max,V_max,
a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
       X_mejor_mejor=X;
    end
       %%Ordenar FSM
    for i=1:SN

f_obj_r_FSM(i)=funcion_obj_ANS(FSM(:,:,i),X_i,V_min,a_min,b_min,a_max,b_m
ax,V_max,a_i,V,a_p,X_ult_7_dias,T,H_T_F,a_p_g,b_p_g);
    end
    [f_obj_r_FSM,I]=sort(f_obj_r_FSM,'descend');%f_obj_r_FSM    son    los
valores f_obj ordenados en orden descendente.
    FSM=FSM(:,:,I);
    K=K(I);
end
%Finaliza exploradora
end
%Finaliza HABC
```

8.3 Apéndice C: Artículo de investigación

MARCO DE TRABAJO PARA LA PROGRAMACIÓN DE TURNOS DE ENFERMERÍA

CHRISTIAN CAMILO MENESES PICO

Ingeniería Industrial

Universidad Industrial de Santander

christianmani2006@gmail.com

Bucaramanga, Colombia

RESUMEN:

El problema de programación de turnos de enfermería (NSP) es un problema multiobjetivo complejo al cual se le ha prestado especial atención en la investigación reciente. La necesidad de mantener al personal satisfecho en su trabajo toma cada vez mayor importancia en el sector salud debido a su impacto directo en la satisfacción de los usuarios del servicio; es por esto que es cada vez mayor la necesidad de programar horarios que reflejen las preferencias de los empleados a la vez que estos horarios sean eficientes en términos de cumplimiento de demanda. En el presente trabajo se desarrolla un modelo para el NSP basado en auto-programación, los resultados muestran que es factible utilizar este tipo de modelo para dar solución al NSP y se obtienen grandes ventajas con respecto al modelo clásico de formulación de NSP. El modelo se resuelve

utilizando dos técnicas modernas de solución: La búsqueda adaptativa en el vecindario (ANS) y el algoritmo híbrido de colonia de abejas (HABC) logran ser eficaces para dar solución al problema formulado con auto-programación. Los resultados de este trabajo muestran que es posible utilizar modelos como la auto-programación para representar más verazmente al problema a la vez que satisfacer las necesidades de demanda crecientes en el entorno actual.

PALABRAS CLAVE: Problema de programación de turnos de enfermería, Investigación de operaciones, Programación de turnos de trabajo.

ABSTRACT:

Nurse Scheduling Problem (NSP) is a complex multiobjective problem wich has received special attention on recent investigations. The need of keeping the staff satisfied in their job takes more importance in the health sector; this is why its important to schedule timetables that reflect the preferences of the employees and also be efficient in terms of demand. In this work a model for NSP is developed based on auto-scheduling, the results show that it is possible to use this type of model for solving the NSP and obtain big advantages over the classical formulation methods. The model is solved using two modern techniques of solution: The adaptative neighborhood search (ANS) and the hybrid of the bee colony algorithm (ABC) show to be effective for solving the problem formulated with auto-scheduling. The results of this work show that it is possible to use models like the auto-scheduling model to represent more truly the problem and at the same time satisfy the bigger current demand of staff.

KEY WORDS: Nurse scheduling problem, Operations Research, Shift scheduling.

Introducción

La programación de enfermeras se relaciona con otros recursos propios de la atención médica y su rol es bastante importante en la provisión de un servicio médico de alta calidad, siendo usualmente las de mayor interacción con los usuarios de este servicio. Una programación deficiente resultará en insatisfacción laboral y una alta rotación de

personal, especialmente en países en vía de desarrollo, la migración de enfermeras hacia otros países con mejores posibilidades hace que el sistema de salud del país en cuestión se vea afectado y exista escasez de talentos (M'Hallah y Alkhabbaz, 2013). Adicionalmente la insatisfacción laboral en prestación de servicios de salud es una consecuencia indeseada debido a los altos riesgos de que existan errores humanos, así como el deterioro del servicio en general.

Un problema de programación de turnos de enfermería consiste en asignar una serie de recursos, en este caso enfermeras, a un conjunto de turnos especificados para un periodo definido. El problema está sujeto a una serie de restricciones tanto duras como blandas y la naturaleza del problema lo hace un problema de tipo NP-hard cuyos objetivos usualmente son múltiples.

Revisión de la literatura

Muchos autores han tratado el problema de NSP desde distintos enfoques de la investigación de operaciones, desde métodos exactos utilizando programación lineal entera, algoritmos branch-and-price, programación de restricciones… entre otros, hasta la aplicación de las conocidas metaheurísticas adaptadas al problema de NSP siendo las más utilizadas el algoritmo genético (GA) y la búsqueda del vecindario variable (VNS). Se ha observado un incremento en los últimos años de los autores que trabajan con técnicas metaheurísticas, métodos híbridos (entre exactos y metaheurísticas) y el mejoramiento de la eficiencia computacional y de calidad de las soluciones para los métodos exactos. También se han introducido las denominadas hiperheurísticas, las cuales utilizan una serie de heurísticas y tienen la cualidad de seleccionar la mejor heurística de solución al problema específico.

Rönnberg y Larsson (2010) describen el modelo denominado auto-scheduling para la programación de los turnos y desarrollan un modelo que lo automatiza. El modelo planteado expresa las restricciones suaves y duras en forma de votos en contra o pedidos para trabajar en ciertos turnos realizados por las enfermeras. Posteriormente es formulado y resuelto por métodos exactos como un problema MIP en el cual un voto limitado para no trabajar en un turno específico es considerado como restricción dura.

Metaheurísticas

Burke, Curtois, Van Draat, Van Ommeren y Post (2011) utilizan un modelo genérico de NSP y centran su estudio en mejorar la búsqueda local iterativa (ILS). Especifican cómo medir el progreso que hace la búsqueda a lo largo del

tiempo: los parámetros para medir esto son el "horizonte de interés" en el que no interesan los valores iniciales de búsqueda sino el progreso que ésta ha realizado, "criterio de reinicio" y "criterio de olvido de reinicio" estos últimos encargados de especificar cuándo parar la medición. Chiaramonte, Cochran y Caswell (2014) utilizan el método de intercambio entre agentes (CNR) para modelar el problema de NSP. En este modelo cada enfermera es una entidad que busca satisfacer al máximo sus preferencias personales "pagando" a cambio con un aporte a la mejora de preferencias de demanda. De esta manera se genera un modelo que busca satisfacer las preferencias de las enfermeras a la vez que las de demanda. Para mejorar el modelo, se combina con una ILS que permite el intercambio entre agentes de asignaciones al horario o "acciones".

Lü y Hao (2012) Introducen el método ANS para su solución, el cual está inspirado en VNS con dos tipos de búsqueda adicionales. Cada tipo de búsqueda varía en intensidad y diversificación y se alterna entre ellas dependiendo del estado de la búsqueda actualmente. Finalmente si no ha habido mejora en la última *n* iteraciones, la búsqueda se reinicia.

Awadallah, Bolaji y Al-Betar (2015) introducen un híbrido entre el algoritmo de colonia de abejas (ABC) y el método de búsqueda intensiva de ascensión en colina (HCO) al NSP. El algoritmo tradicional de ABC es modificado en la fase de la búsqueda de soluciones por las abejas obreras, en esta fase se utiliza el método HCO para encontrar el óptimo local de cada solución almacenada en la memoria utilizando 4 tipos distintos de vecindario.

Mutingi y Mbohwa (2015) desarrollan un modelo Fuzzy para el problema de NSP, las restricciones suaves son evaluadas con funciones de ajuste cada una y la función objetivo es el valor mínimo entre todas estas funciones, adicionalmente tratan los requerimientos de demanda como una restricción suave susceptible de ser penalizada gradualmente dentro de un intervalo definido. Para darle solución introducen el algoritmo de simulación *fuzzy* de metamorfosis (FSM) al NSP; inspirado en el proceso evolutivo biológico que sufren algunos insectos conocido como metamorfosis. Las soluciones son evaluadas con sus funciones de ajuste respectivas y cada violación a las restricciones es medida respecto a una función triangular o una función lineal decreciente; el puntaje de cada solución siendo la función con menor valor. Posteriormente se mejora la solución iterativamente usando heurísticas sencillas de cambio de turnos; la

"hormona de crecimiento" especifica qué elementos (enfermeras o días) son escogidos para mejora de acuerdo a un componente estocástico y al valor de su función de ajuste. Finalmente se le da mayor flexibilidad al FSM al ser posible alterar las soluciones de acuerdo a los distintos "pesos" que se le pueden asignar a cada restricción o conjunto de restricciones, esto permite que el usuario pueda elegir entre darle mayor prioridad a satisfacer a las enfermeras o a cumplir los requerimientos de demanda, o asignarle mayor prioridad a una restricción en particular.

Ásgeirsson (2014) formula el problema de NSP y lo resuelve utilizando heurísticas con una solución inicial formulada por auto-programación. El método consiste en hacer factible la solución inicial propuesta por los mismo empleados, posteriormente se aplica una serie de algoritmos para mejorar la solución, atacando en cada uno una restricción específica.

Formulación del modelo

1. Consideraciones a tener en cuenta seleccionar un modelo NSP

Drake (2014) menciona que existen consideraciones de relaciones humanas y de dinámica de grupos en la programación de turnos de enfermería y que son factores importantes a tener en cuenta en las distintas fases del modelo de solución. Adicionalmente esta es una de las razones por las que la literatura sobre NSP no encaja en la práctica real del sector salud y los algoritmos y modelos planteados por los autores no son recibidos con completa aceptación en la práctica. Por estas razones un modelo de solución, además de cumplir los objetivos de programación y usar la menor cantidad de recursos para lograrlo, también debe ser incluyente de factores sociales y humanos que pueden alterar la representación correcta del modelo.

Mutingi y Mbohwa (2015) también mencionan la importancia de incluir la igualdad de la programación dentro del método de solución, adicionalmente mencionan que las preferencias y las expectativas de los empleados deben ser también consideradas. "Si bien pueden ser imprecisos y conflictivos, estos factores deben ser considerados cuando se construyan horarios de trabajo." (Mutingi & Mbohwa, 2015).

2. Modelo basado en auto-programación

La auto-programación es una forma de programación de personal en la que los mismos empleados son los encargados de generar un horario que satisfaga los requerimientos de la

organización y que a su vez resuelva los conflictos surgidos entre los empleados.

La auto-programación consta básicamente de dos partes: comienza por recibir los horarios planteados por las enfermeras y posteriormente asigna una enfermera para resolver los conflictos surgidos en el horario final; este proceso es llamado de negociación y es necesario que varias enfermeras se pongan de acuerdo sobre múltiples conflictos. Este método puede ser tedioso a medida que aumenta el tamaño del personal y dedica la mayor parte del tiempo a resolver problemas de intereses entre los mismos empleados, lo cual puede afectar el clima laboral y generar disputas entre las enfermeras. Adicionalmente, pueden no lograrse los objetivos de la organización si se omite la fase de negociaciones, ya que la programación va a tender por suplir las necesidades del personal pero no necesariamente asegurar los requerimientos de demanda de la organización.

Para contrarrestar estas desventajas, la aplicación de técnicas metaheurísticas a este método para resolver el problema es una aplicación viable en términos de eficiencia de uso de recursos y se adapta muy bien a los distintos niveles de complejidad del problema.

3. Formulación del modelo

En vista de la revisión de literatura efectuada y las consideraciones a tener en cuenta para modelar el NSP, el presente trabajo plantea una formulación de modelo que utiliza auto-programación y funciones de ajuste para representar la calidad de las soluciones.

3.1 Características del modelo:

3.1.1 Matriz de turnos: La matriz tiene como variable X_{ij}, donde i es la enfermera y j es el día del periodo de programación. Si se toma como ejemplo una instancia con tres turnos de trabajo, X_{ij} puede tomar entonces los valores de cada turno, para un ejemplo de 3 tipos de turno y sus horarios:

Noche, N: Turno de 0:00 a 8:00.

Mañana, M: Turno de 8:00 a 16:00.

Tarde, T: Turno de 16:00 a 0:00.

Fuera, F: Día no programado.

El turno ‘F’ siempre será definido como turno fuera y su letra será reservada.

3.1.2 Definición de las características del modelo:

3.1.2.1 Turnos de trabajo:

Siguiendo el mismo ejemplo de cuatro turnos de trabajo posibles

Noche, N: Turno de 0:00 a 8:00.

Mañana, M: Turno de 8:00 a 16:00.

Tarde, T: Turno de 16:00 a 0:00.

Fuera, F: Día no programado.

3.1.2.2 Características de las enfermeras

Para cada enfermera $i = [1, I]$

Vector $X\_i(i, d)$ con las preferencias de turnos.

Vector lógico $V(i, d)$ con los turnos vetados de cambio.

3.2 Restricciones duras:

3.2.0 Un turno por día: Todas las enfermeras serán programadas con un turno por día solamente. Esta restricción dura siempre se cumple debido a la forma de representar la matriz y a que X_{ij} siempre toma un solo valor.

3.2.1 Secuencia de turnos prohibida: Una enfermera no podrá trabajar en una secuencia de turnos prohibida. Un ejemplo de secuencia de turnos prohibida, siguiendo el mismo ejemplo anterior puede ser T-N ya que el tiempo de descanso entre los dos no es suficiente (nulo en este caso).

3.2.2 Día de descanso obligatorio cada 7 días:

Una enfermera no podrá trabajar más de 6 días consecutivos sin descanso. En este lapso de tiempo debe existir un turno fuera ('F') programado para la enfermera.

3.3 Restricciones suaves:

3.3.1 Funciones de ajuste para restricciones suaves:

La manera de representar las restricciones suaves en el modelo se da por medio de funciones de ajuste (Mutingi y Mbohwa, 2015). Para cada restricción suave existirá una función de ajuste específica que evalúa el nivel de violación de esta restricción; existen dos tipos de funciones de ajuste:

- Función de tipo triangular:

Esta función se encarga de modelar restricciones para valores específicos con una media m deseada y un valor a de desviación máxima permitida (Figura 1)

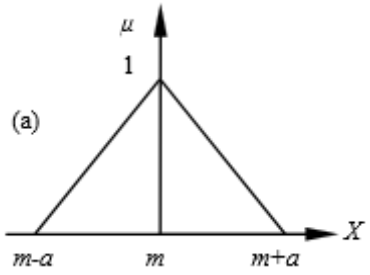


Figura 1: Función de ajuste triangular (fuente: Mutingi y Mbohwa, 2015, p. 4).

- Función linear decreciente

Esta función se utiliza cuando se desea satisfacer un intervalo específico y los valores que no estén dentro del intervalo serán penalizados

conforme se alejen más de éste (Figura 2). La diferencia con la función anterior es que en la función linear se acepta una desviación *a* sin penalizar a la restricción, caso contrario a la función triangular, donde la mínima desviación de la media produce una penalización proporcional.

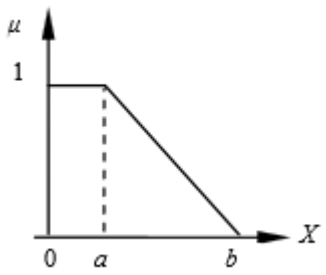


Figura 2: Función de ajuste linear decreciente (fuente: Mutingi y Mbohwa, 2015, p.4.).

3.3.2 Restricciones de demanda:

Las restricciones de demanda se encargan de definir las necesidades del hospital para suplir el nivel de servicio deseado en determinadas horas del día.

Dadas las condiciones realistas del entorno, definir las restricciones de demanda como un valor rígido a ser cumplido puede no ser representativo de la realidad de la programación ya que en algunos casos la restricción dura puede ser violada indiferentemente. Por otro lado, al definirla simplemente como una restricción suave, no se le agrega al modelo la importancia que tiene satisfacer la demanda en los hospitales. Es por esto que la función de ajuste que mejor represente a las restricciones de demanda debe contener un corto intervalo de desviación admitida, acompañado de una alta ponderación con respecto a las demás funciones. La función que mejor se adapta a estos requisitos es la función linear decreciente (Figura 2), con un valor *a* no muy grande que penalice las violaciones grandes a la restricción. Esto permite darle flexibilidad al problema manteniendo el nivel de servicio dentro de las prioridades del modelo. Las funciones serán dos: Una que mida sub-programación y otra que mida sobre-programación.

3.3.3 Igualdad en la programación:

Como se mencionó en la sección 4.4 la igualdad en la programación de turnos debe ser medida y controlada. Para esto se usa una función de ajuste triangular que mide, para cada enfermera, la desviación respecto a la media general de carga laboral en cada periodo.

3.3.4 Preferencias de los empleados:

El apartado de las preferencias de los empleados es complejo debido a que no es muy claro cuales turnos son menos deseados de ser cambiados que otros para cada empleado. Es por esto que se debe emplear un mecanismo que permita definir para cierto número de días de cada

semana cuales turnos son más deseados de no ser modificados. Adicionalmente es lógico asumir que el empleado deseará un horario final similar al que propuso inicialmente; los parámetros para medir esta similitud son diversos y van desde distancia entre turnos para el mismo día hasta la cantidad de tipos de turnos similares en la semana.

El objetivo será entonces minimizar la variación de turnos respecto al horario inicial y adicionar mayor penalización si el turno cambiado es un turno vetado. Los turnos vetados serán almacenados inicialmente en un conjunto *V* para cada periodo, por lo tanto, la función evaluará la cantidad de cambios de turno total realizada por los algoritmos. Adicionalmente se debe procurar satisfacer las preferencias de los empleados de manera equitativa en la matriz de turnos, por lo tanto a esta restricción se le adiciona un componente que mida en la satisfacción de preferencias.

3.4 Parámetros del modelo

3.4.1 Horario

i: Subíndice para enfermeras $[1, I]$

d: Subíndice para día en el periodo de programación $[1, D]$

T:Conjunto de tipos de turno de trabajo

$X(i, d)$: Tipo de turno asignado a la enfermera i en el día de programación d

3.4.2 Demanda

$Vmin(p, d)$: Valor de demanda mínima para el turno p en el día d; $P = \{N, M, T\}$

$Vmax(p, d)$: Valor de demanda máxima para el turno p en el día d; $P = \{N, M, T\}$

3.4.3 Carga laboral

$c(i)$: Carga laboral promedio para la enfermera i

3.4.4 Preferencias

$\{Vi\}$: Conjunto de turnos vetados para la enfermera i

$v(i)$: Variación total de turnos para la enfermera i

3.6 Definición de las funciones de ajuste

3.6.1 Demanda

3.6.2 Disminuir la sub-programación

Siendo $Vmin(p, d)$ el valor mínimo de demanda para el turno p en el día d y $Vreal(p, d)$ el valor actual obtenido, entonces

$$X_d = \sum_{p=1}^{P} Vmin(p, d) - Vreal(p, d)$$

La función de ajuste a utilizar es la función linear decreciente

$$\mu_1(Xd)$$

Donde a (faltantes)es la tolerancia para sub-programación y b el límite para el intervalo de demanda

3.6.3 Disminuir la sobre-programación

Siendo $Vmax(p,d)$ el valor máximo de demanda para el turno p en el día d y $Vreal(p,d)$ el valor actual obtenido, entonces

$$Xd = \sum_{p=1}^{P} Vreal(p,d) - Vmax(p,d)$$

La función de ajuste a utilizar es la función linear decreciente

$$\mu_2(Xd)$$

Donde a es la tolerancia para sobre-programación (sobrantes) y b el límite para el intervalo de demanda

3.6.4 Igualdad en la programación

Siendo *c(i)* el nivel de programación de la enfermera *i*, entonces

$Xi = ci - \beta.$

donde β es el promedio de carga laboral en el periodo.

Se utiliza la función de ajuste triangular

$$\mu_3(Xi)$$

Donde a es el intervalo de desviación y m= β

3.6.5 Preferencias de los empleados

Siendo *v(i)* la variación de turnos del empleado i :

$$v_i = \beta * V_i + (1-\beta) * Vp_i$$

Donde β es la penalización por cambiar un turno vetado, V_i es la cantidad de cambios de turnos vetados y Vp_i es la cantidad de veces que un turno normal es cambiado.

Entonces, la función de ajuste para esta restricción será:

$$\mu_4(Xi) = \mu_{4.1}(Xi_1) * p1 + \mu_{4.2}(Xi_2) * p2$$

Donde

$$Xi_1 = v(i) - \gamma$$

representa la desviación respecto a la media de satisfacción total para todas las enfermeras, siendo γ el promedio para el valor de desviaciones en el periodo. Se utiliza una función de ajuste de tipo triangular para pasar de Xi_1 a $\mu(Xi_1)$:

Donde a es el porcentaje de desviación respecto a la media (m) permitido y $m = \gamma$.

y

$$Xi_2 = v(i)$$

Xi_2 mide simplemente el nivel de satisfacción de cada enfermera y se utiliza una función de ajuste de tipo linear decreciente para transformar este valor en $\mu(Xi_2)$:

Donde a es el nivel mínimo de insatisfacción y b es el valor máximo aceptado.

Finalmente, ambas funciones de ajuste son ponderadas para dar un valor total a μ_4 . $p1$ y $p2$ siendo los valores de ponderación de $\mu_{4.1}$ y $\mu_{4.2}$, respectivamente.

3.7 Función objetivo

La función objetivo que mide la calidad del horario de programación está definida como sigue:

$$f = \mu_1 + \mu_2 + \mu_3 + \mu_4 - pen_{H1} - pen_{H2}$$

Donde μ_1, μ_2, μ_3 y μ_4 son las funciones de ajuste para las restricciones suaves de sub-programación, sobre-programación, igualdad de carga laboral y satisfacción de las preferencias. pen_{H1} y pen_{H2} son valores negativos de gran magnitud que miden la violación a la restricción dura #1: secuencia prohibida y a la restricción dura #2: día de descanso.

4. Método de solución

Se propone resolver el modelo planteado en la sección cuatro utilizando un método de solución inspirado en la búsqueda adaptativa ANS y posteriormente se resuelve en un marco de HABC.

4.1 Búsqueda adaptativa en el vecindario (ANS)

La búsqueda adaptativa alterna entre tres tipos de búsqueda en vecindario con distintas intensidades, el criterio de alternancia entre estas búsquedas es un valor *dl* que incrementa o disminuye dependiendo de si la función objetivo mejora o empeora. El valor de la función objetivo será entonces la suma de los valores asignados al nivel de satisfacción de cada restricción y el objetivo será aumentar este valor lo máximo posible.

4.1.1 Tipos de movimiento utilizados:

Para explorar el vecindario, en cualquiera de las fases se utilizan cuatro tipos posibles de movimientos:

- ρ_1: Cambiar turnos entre dos enfermeras del mismo día, una de ellas con turno Fuera.
- ρ_2: Cambiar turnos entre dos enfermeras del mismo día, ambas con turno ≠ Fuera.
- ρ_3: Cambiar turnos entre dos días para la misma enfermera, un día de estos con turno Fuera.
- ρ_4: Cambiar turnos entre dos días para la misma enfermera, ambos turnos ≠ Fuera.

En cada iteración el movimiento escogido es aleatorio y depende de q

Donde $q = 1 - \varphi * dens$

- Si $random[0,1] < q$ se escoge el movimiento ρ_1 o ρ_3 con una probabilidad de 0.4 y 0.6 respectivamente.
- Si $random[0,1] > q$ se escoge el movimiento ρ_2 o ρ_4 con una probabilidad de 0.4 y 0.6 respectivamente.

Donde $random[0,1]$ es un número aleatorio entre 0 y 1.

4.1.2 Tipos de búsqueda:

El parámetro dl controla el tipo de búsqueda que efectúa el algoritmo en una iteración dada, a continuación se mencionan los tres tipos de búsqueda en orden de menor a mayor diversificación y se describe el procedimiento de cada una:

- Búsqueda intensiva: La búsqueda intensiva tiene la particularidad de utilizar lista tabú, debido a que es la que mayor intensificación aporta, se utiliza una lista tabú para no repetir movimientos y progresar más rápidamente en esta etapa. A continuación se describe el procedimiento:
    1. Aplicar movimiento ρ con lista tabú.
    2. Lista tabú incluye los movimientos recíprocos por los siguientes $tl + rand * 3$ movimientos. $(tl = 0.8 * E)$ donde E es el número total de enfermeras.
    3. Escoge la mejor solución. Si mejor solución es Tabú se escoge siempre y cuando mejore la solución actual.
- Búsqueda intermedia: La búsqueda intermedia se utiliza cuando se desea un nivel de diversificación mayor que la búsqueda intermedia. En esta fase, la búsqueda se limita a la mitad de la matriz X_{ij}, el procedimiento es el siguiente:
    1. Limita la búsqueda a un set $S*$ de enfermeras. $S* = E/2$.
    2. Aplicar movimiento ρ
    3. Escoger la mejor solución.
    4. Si la mejor solución es el movimiento más reciente se escoge la segunda mejor con probabilidad de 0.5.
- Búsqueda diversificada : Finalmente si se desea un nivel mayor de diversificación se utiliza la búsqueda diversificada; esta además de restringir la búsqueda a la mitad de la matriz, escoge un movimiento solamente con el criterio de si mejora o no una sola restricción, sin importar que empeore a las demás. Esta fase aporta una diversificación mayor a la búsqueda ya que, si bien las soluciones en esta etapa pueden no ser mejores, puede ayudar a escapar de mínimos locales.
    1. Se escoge un set S* de enfermeras. $S* = E/2$.

2. Se identifican grupos de movimientos sub-prometedores (si por lo menos movimiento mejora una de las restricciones blandas).
3. La solución se escoge al azar entre los movimientos sub-prometedores si los hay. Si no existe ninguno, se escoge cualquier solución del set $S*$.

Criterio de alternancia

El criterio para usar una u otra búsqueda se da con el parámetro dl, el cual mide el nivel de diversificación necesaria en la búsqueda actual.

- Si dl $\in [0, \beta_1)$ →Aplicar búsqueda intensiva.
- Si dl $\in [\beta_1, \beta_2)$ →Aplicar búsqueda intermedia.
- Si dl $\in [\beta_2, 1)$ →Aplicar búsqueda diversificada.

8.3.1.1 4.1.3 Dinámica de dl:

Si la función objetivo no ha mejorado en los últimos θ pasos: $\theta = \frac{E*T}{10}$, siendo E el número total de enfermeras y T el número de tipos de turno (sin incluir turno Fuera) el parámetro dl aumenta como sigue:

$$dl \leftarrow dl + \frac{1 - dl}{6}$$

Si por el contrario la función objetivo mejora en cualquiera momento, el parámetro dl disminuye:

$$dl \leftarrow dl - \frac{dl}{10}$$

8.3.1.2 4.1.3 Criterio de reinicio

Si no hay mejora en cierto número R de iteraciones se reinicia la búsqueda desde la mejor solución X encontrada o desde la solución X actual, con una probabilidad de 0.5 para cada opción. Adicionalmente el parámetro dl se hace 1 sin importar su valor actual. ($dl = 1$).

Método de solución: HABC

4.2 Búsqueda HABC

La búsqueda HABC fue propuesta inicialmente por (Bolaji, Al-Betar & Awadallah, 2013) .Es un híbrido entre la metaheurística de la búsqueda de colonia de abejas (ABC) y el algoritmo de optimización en colina (HCO).

4.2.1 Algoritmo de colonia de abejas (ABC)

El ABC se ha utilizado con éxito en la programación de turnos de personal de diversas áreas (Awadallah, Bolaji & Al-Betar, 2015). Es un algoritmo evolutivo y poblacional que hace uso de operadores no determinísticos para mejorar

iterativamente las soluciones presentes en la población, hasta alcanzar un criterio de parada. Las etapas de búsqueda del ABC se asemejan a una población de abejas en su búsqueda por alimento: Existen tres etapas y por lo tanto tres tipos de abejas en ABC: Las abejas obreras, las abejas observadoras y las abejas exploradoras. Cada tipo de abeja tiene una función específica dentro de la búsqueda por la mejor solución.

4.2.2 Algoritmo híbrido ABC con HCO: Para mejorar el desempeño de las metaheurísticas, varios híbridos con otras búsquedas han resultado en mayor eficacia, a cambio de un aumento en el tiempo de procesamiento. El algoritmo de ascensión en colina (HCO) ha demostrado que puede resultar efectivo en su hibridización con otras búsquedas metaheurísticas poblacionales (Awadallah, Bolaji & Al-Betar, 2015); en el caso del ABC y su híbrido con HCO (HABC) éste fue utilizado para resolver el problema de programación de cursos universitarios (Bolaji, Hkader, Al-Betar & Awadallah, 2014) inicialmente y posteriormente aplicado al NSP (Awadallah, Bolaji & AL-Betar, 2015). El objetivo de esta mejora es mejorar la fase intensificadora del ABC al reemplazar el procedimiento común de las abejas obreras por el método HCO.

En este método denominado HABC, las abejas obreras utilizan el HCO para encontrar el óptimo local de cada solución presente en la población. Esto conlleva a mayor uso computacional a cambio de un aumento en la intensificación del algoritmo.

4.2.2.1 Tipos de movimiento: Los movimientos (ρ) utilizados por las abejas obreras y las abejas observadoras son los mismos que los utilizados en la técnica ANS (Ver sección 5.1.1: Tipos de movimientos).

8.3.1.3 4.2.2.2 Algoritmo HABC

La búsqueda HABC se da como sigue:

1. Iniciación de los parámetros poblacionales:
   - SN: Tamaño de la población
   - MCN: Número máximo de iteraciones
   - $limite$: Número máximo de iteraciones sin mejora, criterio de exploración.
   - HCR: Probabilidad de escogencia para HCO ([0,1])
2. Generación de FSN inicial (Población inicial)
   - Generar diez soluciones iniciales por medio de rankeo de turnos: Esto consiste en ordenar de menor a mayor ‘dificultad’ de programación los turnos a ser

programados. El criterio para ordenarlos, es decir la dificultad de cada turno, se mide por el número total de turnos necesarios en el periodo. Por lo tanto el turno con mayor dificultad es aquel que tenga la menor cantidad de turnos demandados. El método de rankeo de turnos se encarga de crear una solución inicial al asignar enfermeras al azar a los turnos siguiendo su orden de dificultad, comenzando por el que tenga mayor dificultad. El resultado es una solución con enfermeras asignadas al azar a una serie de turnos, 'llenando' primero los de mayor dificultad de programación.

- Ordenar la población inicial de mayor a menor valor calidad de función objetivo.

3. Mejora de cada solución por abejas obreras utilizando HCO.
   - Seleccionar una solución de la población, comenzando por la primera.
   - Comparar parámetro HCR con $random[0,1]$. Si $HCR > random$ efectuar HCO sobre la solución escogida en 1. HCO consiste en efectuar movimiento ρ hasta alcanzar un mínimo local.
   - Pasar a la siguiente solución y repetir paso 2 hasta alcanzar todas las soluciones de la población.
   - Ordenar soluciones de mayor a menor valor calidad de función objetivo.
4. Escogencia por probabilidad proporcional de una solución en FSN para mejora por abeja observadora.
   - Calcular probabilidad acumulada para cada solución de la población.
   - Evaluar la probabilidad acumulada ($prob\_acum$) de cada solución:
   - Escoger la primera solución
   - Si 1-$prob\_acum$ <= $random[0,1]$ Escoger la solución para mejora por abeja observadora. Si no, continuar hasta encontrar solución que satisfaga desigualdad.
   - Aplicar movimiento ρ a solución escogida para mejora.
   - Ordenar soluciones de mayor a menor valor calidad de función objetivo.
5. Escogencia de solución estancada para diversificación por abeja exploradora.
   - Se selecciona una solución s al azar entre la población.
   - Se evalúa el número de iteraciones que han pasado desde que solución s ha percibido una mejora ($pruebas$). Este valor se compara con $limite$.

- Si $pruebas \geq limite$ se reemplaza la solución escogida por una solución aleatoria y se actualiza la población de soluciones. El algoritmo almacena en su memoria la información de la mejor solución encontrada en todo momento.
- Repetir 2 a 5 hasta alcanzar criterio de parada (MSN).

El resultado de la búsqueda es la mejor solución encontrada por el algoritmo.

5.Resultados experimentales

5.1 Resultados ANS

El modelo planteado fue resuelto por la técnica ANS sobre tres instancias de la literatura (Gpost, Ortec y España) . A continuación se muestran los resultados:

9.1.1 5.1.1 Problema inicial: asignar cantidad global de turnos:

De acuerdo a la experimentación inicial se descubrió que es necesario modificar la búsqueda ANS complementándola previamente con otra búsqueda que resuelva el problema inicial de cantidad de turnos totales de cada tipo, descrito a continuación.

La programación de turnos de enfermería utilizando los movimientos planteados (para explorar el vecindario) en ambos métodos de solución generan un problema nuevo: El problema inicial de asignar la cantidad global de turnos de cada tipo, incluidos los turnos fuera. Esto debido a que los movimientos posibles solamente alternan turnos entre turnos existentes y no eliminan o crean ninguno nuevo. Por lo tanto la búsqueda de una solución para equilibrar los deseos de demanda y de satisfacción de preferencias estará sesgada hacia esta última si no se estudia inicialmente la cantidad de turnos de cada tipo (incluido el turno fuera):

Solución :

Se utiliza una búsqueda sencilla en el vecindario seleccionando en cada iteración el movimiento que tenga mayor valor en términos de función objetivo. El movimiento mencionado consiste en cambiar un turno de la matriz actual por un turno diferente y evaluar el puntaje de X con este cambio.

5.1.2 Resultados Gpost

Esta instancia consiste en una cuadrilla de 6

V_min, V_max :	Día 1	Día 2	Día 3	…	Día 28
D	3	3	3	…	3
N	1	1	1	…	1

enfermeras distribuidas en un periodo de programación de 28 días. Existen solo dos tipos de turno: día (D) y noche (N), no existe secuencia prohibida de los mismos, por lo que en esta instancia no aplica la restricción dura #1.

La demanda mínima y máxima es de 3 turnos tipo 'D' y 1 turno tipo 'N' para cada día de programación. Los datos se resumen en la tabla 1:

Tabla 1: Valores de demanda para Gpost

En la ilustración 1 se muestra la matriz inicial auto-programada, el número total de turnos asignados en la matriz (incluyendo turnos fuera) es de 168 turnos los cuales están distribuidos como se muestra en la tabla 2.

Tabla 2: Distribución de turnos asignados en matriz inicial

Número de turnos asignados tipo 'D'	**57**
Número de turnos asignados tipo 'N'	61
Número de turnos asignados tipo 'F'	50
Número total de turnos asignados	168

9.1.1.1 5.1.2.1 Búsqueda inicial

Tabla 3: Número de turnos por tipo después de búsqueda inicial

Número de turnos asignados tipo 'D'	**84**
Número de turnos asignados tipo 'N'	28
Número de turnos asignados tipo 'F'	56
Número total de turnos asignados	168

La cantidad de turnos 'D' y 'N' fue modificada para satisfacer mejor los requerimientos de demanda. La cantidad restante de turnos es asignada a turnos tipos 'F': (ver tabla 3).

9.1.1.2

9.1.1.35.1.2.2 Búsqueda ANS:

Una vez resuelto el problema inicial, se procede a mejorar el horario por medio de la técnica ANS (Búsqueda adaptativa). Se programaron un máximo de 1500 iteraciones con los siguientes parámetros de búsqueda:

Tabla 4: Parámetros de búsqueda ANS

# de iteraciones	**1500**
Betha_1	0.3
Betha_2	0.5

Juntando las dos búsquedas (Inicial y ANS) la función objetivo pasó de 40.9271 a 65.2514, produciendo el mayor puntaje posible en los valores de ajuste de las restricciones de demanda (28), los datos completos se pueden observar en la tabla 5.

Tabla 5: Resultados Gpost

	Inicial	Final
Restricción suave #1	19.5	28
Restricción suave #2	18	28
Restricción suave #3	0.6271	4.5714
Restricción suave #4	6	4.68
Restricción dura #2	3.2	0
Función objetivo	40.9271	65.2514

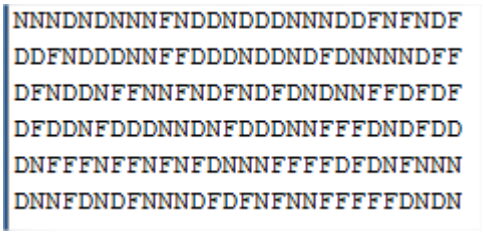


Figura 3: Matriz inicial Gpost

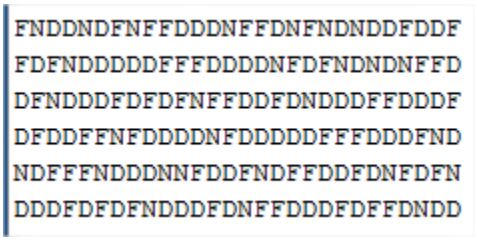


Figura 4: Matriz final Gpost

5.1.3 Resultados Ortec

La instancia Ortec consta de 13 enfermeras, para ser repartidas en 28 días de programación. Existen tres tipos de turno: 'D','E','L' y 'N'. Las secuencias prohibidas de turno se resumen en la tabla 4:

Tabla 6: Secuencia de turnos prohibida Ortec

L-E
N-D
N-E
N-L

La demanda mínima y máxima deseada en el periodo es de:

- 1 turno 'N' todos los días.
- 3 turnos 'D', 3 turnos 'E' y 3 turnos 'L' todos los días, exceptuando los días del periodo múltiplos de 6 y 7, los cuales requieren solamente 2 turnos de cada uno ('D','E' y 'L').

En esta instancia se le da mayor importancia a satisfacer las restricciones de demanda (Restricciones suaves 1 y 2) y una menor importancia a las restricciones de igualdad y preferencias (Restricciones suaves 3 y 4). Los

anteriores datos se introdujeron al algoritmo (Inicial + ANS) con los mismos parámetros de búsqueda mostrados en la tabla 4.

9.1.1.4 5.1.3.2 Búsqueda ANS

Después de ambas búsquedas (Inicial y ANS) la función objetivo pasó de -10.8569 a 77.4146, los resultados se muestran en la tabla 7:

Tabla 7: Resultados Ortec

	Inicial	Final
Restricción suave #1	22.4219	27.7813
Restricción suave #2	17.5	27.7813
Restricción suave #3	6.6213	11.1250
Restricción suave #4	13	10.7271
Restricción dura #1	57.6	0
Restricción dura #2	12.8	0
Función objetivo	-10.8569	77.4146

La matriz de turnos final se muestra en la figura 5:

```
EELFDEFLDLFEENNFLLFDEFDEDLFF
NFEEDDFDEEFEFDLLLFLFNFDLNFDE
LLFLFLFLLDEEEFEFELDLLFFLFEEE
ENFLFEEENNFLFEDLFEFFFELFELDL
LLDEDEFDNFEDDFENNNNNFEFEFFEF
FENFFFDDLFDDFEFEEDEFDLNFLDLL
DLDDNFFEDLNFDDLFFLFEFDLDENNF
LDLFEDEFEFELLFLLFEDDFDEDLFFF
DFLDLFDEDLDDFLFEDDLFLDEFEDFF
EDEDLLFLFDDFLLDEDFDEFNFDLLFF
FFDLLFLFLDLLFFDDEEELFLDLDDFD
FEFFENNNFELFFFEDDDLFDEFEDELD
DDEEEFLFEELNNFFDLFEFELLNFEFN
```

Figura 5: Matriz final Ortec

5.1.4 Resultados España

La instancia España consiste en asignar 16 enfermeras en un periodo de programación de 31 días. Los tipos de turno son: ‘M’, ‘A’, ‘N’, ‘D’ y ‘H’. Las restricciones de secuencia de turnos se muestran en la tabla 8.

- La demanda requerida por día en el periodo de programación es de 4 turnos tipo ‘M’, 4 turnos tipo ‘A’, 2 turnos tipo ‘N’, 1 turno tipo ‘D’ y 4 turnos tipo ‘H’ solamente en los días múltiplos de 5 y 6 del periodo de programación.
- Esta instancia le da mayor importancia a cumplir la restricción suave #1, en orden de importancia le siguen las restricciones 2 y 3 y por último, la de menor importancia en este caso, es la restricción #4. En

otras palabras evitar la sub-programación tiene una prioridad de 1, evitar la sobre-programación y aumentar la igualdad tienen ambas prioridad de 2, y por último satisfacer las preferencias de los empleados tiene una prioridad de 3.

Tabla 8: Secuencia prohibida de turnos instancia España

N-M
N-A
N-D
N-H
D-M
D-A
D-D
D-H
H-M
H-A
H-D
H-H

5.1.4.2 Búsqueda ANS

Aplicando la búsqueda inicial + ANS se obtuvo resultados mostrados en la tabla 9:

Tabla 9: Resultados instancia España

	Inicial	**Final**
Restricción suave #1	21.7082	29.6021
Restricción suave #2	17.1034	30.5889
Restricción suave #3	6.5325	15.3799
Restricción suave #4	16	9.12
Restricción dura #1	126	0
Restricción dura #2	26	0
Función objetivo	-90.6559	84.6909

La matriz de turnos final se muestra en la figura 6:

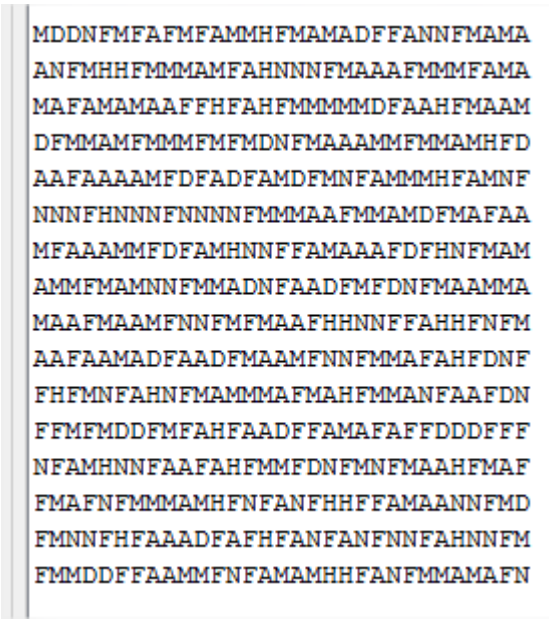


Figura 6: Matriz final España

5.2 Resultados HABC

Al implementar la técnica de solución HABC el recurso computacional aumentó considerablemente, esto debido a varios motivos:

- La técnica de solución es poblacional, por lo que, a diferencia de ANS, HABC mejora iterativamente diez soluciones al tiempo. Considerando esto solamente, el tiempo de cálculo de una iteración aumenta diez veces más.
- El algoritmo HCO utilizado en la etapa obrera de HABC es más exigente computacionalmente que la búsqueda obrera tradicional en el ABC ya que mejora cada solución a su máximo local. Esto combinado con la característica poblacional del algoritmo hacen que el tiempo de cómputo para cada iteración aumente considerablemente con respecto al ANS.

Tomando en cuenta estas consideraciones se evaluaron las instancias Gpost y Ortec aplicando el HABC con 50 iteraciones como límite de parada.

5.2.1 Resultados Gpost

La función objetivo pasó de 40.9271 a 64.1841, produciendo el mayor puntaje posible en el valor de ajuste de la restricción de demanda #1 (sub-programación), los datos completos se pueden observar en la tabla 10. Se corrieron 50 iteraciones del algoritmo.

Tabla 10: Resultados HABC instancia Gpost

Resultados Gpost	**Inicial**	**Final**
Restricción suave #1	19.5	28
Restricción suave #2	18	26.5
Restricción suave #3	0.6271	4.6441
Restricción suave #4	6	5.040
Restricción dura #2	3.2	0
Función objetivo	40.9271	64.1841

Figura 8: Matriz final Gpost

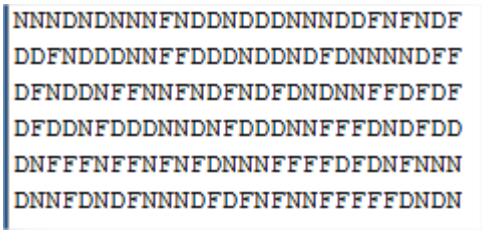


Figura 7: Matriz inicial Gpost

5.2.2 Resultados Ortec

Después de aplicar la búsqueda HABC la función objetivo pasó de -10.8569 a 69.6626. Se corrieron 50 iteraciones del algoritmo. En la siguiente tabla se muestran los resultados por restricción:

Tabla 11: Resultados HABC instancia Ortec

Tabla 3: Resultados Ortec	**Inicial**	**Final**
Restricción suave #1	22.4219	26.6875
Restricción suave #2	17.5	27.7656
Restricción suave #3	6.6213	11.5382
Restricción suave #4	13	9.6713
Restricción dura #1	57.6	0
Restricción dura #2	12.8	0
Función objetivo	-10.8569	69.6626

La matriz de turnos final se muestra en la figura 9:

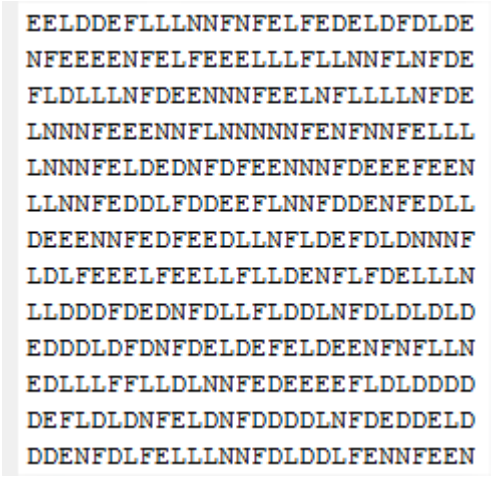


Figura 9: Matriz final Ortec

Conclusiones

- De acuerdo a la revisión de la literatura efectuada sobre el NSP se encontró que un problema común entre los investigadores es la adaptación de las técnicas de solución a las distintas formulaciones del modelo, debido a la gran variedad de restricciones presentes entre un problema y otro. Si bien se evidenciaron esfuerzos en la literatura para crear formulaciones de modelo

genéricas, estas no son suficientes para abarcar la totalidad de variedad en términos de restricciones especialmente de preferencias de empleados.

- De acuerdo a la revisión de literatura se encontró que la robustez de una técnica de solución para NSP de ve disminuida debido a la incorporación de restricciones específicas que limitan o dificultan la aplicación de la técnica en otros modelos diferentes a los planteados.
- El modelo planteado en este trabajo fue programado y probado en MATLAB con una serie de instancias de la literatura que variaban entre sí en términos de tamaño y complejidad, mostrando resultados satisfactorios.
- El método demostró ser posible de ser implementado en un marco de solución de técnicas modernas al problema NSP (ANS, HABC) cuya creación inicial fue basada en modelos con métodos tradicionales de enumeración de restricciones.
- Comparado con los modelos tradicionales, la utilización del método de auto-programación y la utilización de solo dos tipos de funciones de ajuste para representar la calidad del horario en cualquier momento permitieron utilizar el modelo para evaluar una variedad de instancias en tamaño y complejidad, sin necesidad de programar restricciones adicionales de acuerdo a cada problema.
- Para las instancias evaluadas, el método de búsqueda adaptativa en el vecindario (ANS) resulta en una mayor eficiencia computacional que la búsqueda poblacional híbrida HABC.
- En los modelos tradicionales las restricciones de satisfacción de empleados son enumeradas por el programador o investigador, por lo que estas restricciones pueden no reflejar las preferencias de todas las enfermeras. Utilizando el modelo de formulación del problema expuesto en este trabajo se logra aumentar la representación de las preferencias reales de cada enfermera en cada periodo de programación. El resultado es un modelo más sencillo en términos de cantidad de restricciones y una representación más cercana a las preferencias reales de las enfermeras en cada periodo.

- Con la realización de este trabajo se muestra que es posible resolver un problema de NSP con una mayor inclusión de los empleados y a la vez una mayor representación de sus preferencias en los modelos de solución, sin alterar las necesidades de demanda.
- El modelo completo aquí desarrollado puede ser utilizado fácilmente, reemplazando a la técnica manual de

programación, en múltiples escenarios con NSP debido a su flexibilidad y robustez.

Bibliografía

M'Halla H, R. y Alkhabbaz, A. (2013). Scheduling of nurses: A case study of a Kuwaiti health care unit. *Operations Research for Health Care. 2*(1-2), 1–19. http://doi.org/10.1016/j.orhc.2013.03.003

Rönnberg, E. y Larsson, T. (2010). Automating the self-scheduling process of nurses in Swedish healthcare: A pilot study. *Health Care Management Science. 13*(1), 35–53. http://doi.org/10.1007/s10729-009-9107-x

Burke, E. K., Curtois, T., van Draat, LF, Ommeren, J-K. y Post, G. (2011) Progress control in iterated local search for nurse rostering. *The Journal of the Operational Research Society, 62*(2), 360–367. http://doi.org/10.1057/jors.2010.86

Chiaramonte, M., Cochran, D.,. y Caswell, D. (2014). Nurse preference rostering using agents and iterated local search. *Annals of Operations Research, 226*(1), p 443–461. http://doi.org/10.1007/s10479-014-1701-8

Lü, Z., y Hao, J. K. (2012) Adaptive neighborhood search for nurse rostering. *European Journal of Operational Research*. 218(3), 865–876. http://doi.org/10.1016/j.ejor.2011.12.016

Awadallah, M. A.; Bolaji, A. L. A. y Al-betar, M. A. (2015) A hybrid artificial bee colony for a nurse rostering problem. *Applied Soft Computing Journal, 35*, 726–739. http://doi.org/10.1016/j.asoc.2015.07.004

Mutingi, M., & Mbohwa, C. (2015). A multi-criteria approach for nurse scheduling fuzzy simulated metamorphosis algorithm approach. *Mutingi, M., & Mbohwa, C , IEOM 2015 - 5th International Conference on Industrial Engineering and Operations Management* http://doi.org/10.1109/IEOM.2015.7093904

Ásgeirsson, E. I. (2014) bridging the gap between self schedules and feasible schedules in staff scheduling. *annals of operations research*, *218(1)*, 51-69. http://doi.org/10.1007/s10479-012-1060-2

Drake, R. G. (2014). The nurse rostering problem: From operational research to organizational reality? *Journal of Advanced Nursing, 70*(4), 800–810. http://doi.org/10.1111/jan.12238