Localización de Suministros en Logística Humanitaria

Informe de Proyecto de Grado

Instituto de Computación

Facultad de Ingeniería

Universidad de la República

Rolando Lescano

Oscar Muñoz

Tutores

Pedro Piñeyro

Omar Viera

Marzo 2017

Resumen

Cada año alrededor de 500 desastres naturales provocan la muerte de aproximadamente 70000 personas, y afectan a más de 200 millones de personas en todo el mundo. Cuando esos eventos ocurren, se requieren cantidades muy altas de suministros para proveer apoyo a las personas afectadas. Una forma en que las organizaciones que brindan asistencia ante desastres mejoran su efectividad es mediante la localización anticipada de suministros, lo que se conoce como preposicionamiento. Este tipo de problema es difícil de resolver computacionalmente ya que está relacionado a los problemas de localización y cubrimiento, los cuales se conocen por ser NP-hard.

El objetivo del presente proyecto fue desarrollar y evaluar un algoritmo para maximizar el cubrimiento de la demanda de las personas afectadas por un desastre natural, minimizando a su vez los costos involucrados en la localización de suministros. Para ello se utilizó una formulación de programación Bi-Nivel para resolver el problema mencionado anteriormente, tomando como base el modelo *Maximal Covering Location Problem*. Una de las dificultades de la programación bi-nivel es que no existe una forma de hallar una solución óptima de forma eficiente, por lo tanto, lo que se buscó en este trabajo es desarrollar un procedimiento heurístico para intentar obtener una solución con la mayor cobertura posible y de menor costo. Se desarrolló un procedimiento iterativo que utiliza la metaheurística Tabu Search para resolver el problema de primer nivel, y el segundo nivel a través de GLPK.

Se implementaron distintas variantes del algoritmo que emplean diferentes optimizaciones para el problema de primer nivel, consiguiendo así soluciones de distintas características para una misma instancia. La evaluación de la mejor solución se hizo en base al concepto de Costo-Efectividad, la cual considera tanto el costo como el cubrimiento de una solución. Por lo tanto, partiendo de soluciones en las cuales no se tiene en cuenta el costo, se fue evolucionando hasta llegar a una solución de mejor Costo-Efectividad. Los resultados obtenidos mejoraron el Costo-Efectividad obtenido por un algoritmo existente, esto quiere decir que el preposicionamiento es más efectivo en cuanto a costos.

Contenido

1. Introducción

Cada año alrededor de 500 desastres naturales matan aproximadamente 70000 personas, y afectan a más de 200 millones de personas en todo el mundo. Cuando esos eventos ocurren, se requieren cantidades muy altas de suministros para proveer apoyo a las personas afectadas. Hay muchas organizaciones que se encargan de proveer apoyo humanitario tanto de desastres naturales como causados por el hombre, como por ejemplo *CARE International*, la cual es una de las más grandes. Los problemas vitales al tratar con este tipo de desastres son la agilidad en movilizar los suministros y la efectividad en la distribución de los mismos (Duran et al., 2011) [1].

Los desastres naturales son fenómenos que pueden arrasar con los bienes humanos y con la vida misma. Según EM-DAT (Emergency Events Database 2011, base de datos de desastres mundiales) los desastres naturales irán aumentando hasta 5 veces en cantidad y severidad en los próximos 50 años. Entre 1992-2012, fueron afectados 4.4 billones y muerto 1.3 millones, ocasionando una pérdida de 3 trillones de dólares (UNISDR 2012).

Según la Cruz Roja y la Media Luna Roja, se define desastre natural como: Eventos calamitosos que interrumpen la actividad de una sociedad o comunidad y causan pérdidas humanas, materiales, ambientales y económicas que exceden la capacidad de recuperación de la sociedad o comunidad usando solo sus propios recursos (Leiras et al, 2014) [2].

Se puede definir brevemente Logística Humanitaria como el proceso que involucra la movilización de personas y recursos, incluyendo también habilidades y conocimientos para ayudar a las personas afectadas por el desastre (Cozzolino, 2007) [3].

Cuando ocurre un desastre, la falta de suministros o la lenta movilización de los mismos pueden hacer que la respuesta al desastre sea menos efectiva. Una forma en que las organizaciones mejoran su efectividad es mediante el preposicionamiento de suministros ya que a la hora de la emergencia, se ahorraría la parte en la que los suministros deben ser movilizados. Sin embargo, configurar una red de preposicionamiento no es sencillo (Duran et al., 2011) [1].

El objetivo de este trabajo es formular matemáticamente un problema de localización de suministros y un algoritmo para solucionarlo, para lo que se toma como base el modelo matemático y el procedimiento heurístico propuesto por Lee & Lee (2010) [17], en dónde dada una cantidad de personas, y una cantidad de centros de ayuda, se busca maximizar el cubrimiento de los distintos servicios que existen en el problema. El objetivo principal es poder

realizar cambios en dicho modelo matemático, y proponer distintos métodos de resolución, tomando como enfoque principal la noción de costos.

Usualmente el término desastre se refiere a “interrupción que afecta físicamente a un sistema en su conjunto y amenaza sus prioridades y metas“(A. Cozzolino, 2007) [3].

Los esfuerzos humanitarios se organizan a lo largo de dos grandes líneas: (A. Cozzolino, 2007) [3]

1. Alivio de desastres (Acciones destructivas, calamidades y plagas).
2. Trabajo de ayuda continua (principalmente es requerido en crisis y plagas).

El Alivio de desastres es el más costoso (a lo que refiere de logística), con cerca del 80% del costo total dedicado a esa línea, siendo muy importante la inversión en esta área (ya que de ahí resultará en el éxito de la operación o el fracaso del mismo) (A. Cozzolino, 2007) [3].

La Gestión del Desastre es descrita como un proceso compuesto por varias etapas o fases: mitigación, preparación, respuesta y recuperación. Estas etapas constituyen el ciclo de vida de la gestión del desastre. Para la Logística Humanitaria y la cadena de suministros, el proceso involucra las últimas 3 etapas (A. Cozzolino, 2007) [3].

La mitigación refiere a las leyes y mecanismos que reducen la vulnerabilidad social. La fase de preparación se da lugar antes del desastre, y es crucial para prepararse y responder de forma rápida ante el desastre. Por lo que es importante el diseño de redes físicas, sistemas de comunicación e información. Esta fase también usa el conocimiento existente, para mejorar los nuevos métodos.

La fase de respuesta es inmediatamente después del desastre y tiene dos objetivos principales: responder activando “red silenciosa” o “red temporal”. Se le denomina red temporal a la red creada una vez que el desastre ocurre, es un proyecto temporal que debe vincular una o más redes permanentes (Jahre et al., 2009) [4], el segundo objetivo es recuperar en el menor tiempo posible los servicios básicos (respuesta inmediata) y entregar bienes a la mayor cantidad de personas afectadas posible (restauración). Esta fase merece mucha atención para la coordinación y colaboración entre los diferentes actores involucrados en la emergencia. Si bien en la primera fase se comunican con los donadores, ONG entre otros, no es hasta el momento del desastre que se “activa”. La fase de reconstrucción involucra rehabilitación, pensada en largo plazo ya que posiblemente las secuelas se mantienen también a largo plazo.

Las tareas humanitarias acompañan el ciclo de vida de un desastre, incluyendo la preparación, respuesta y recuperación. La habilidad de conducir eficientemente dichas operaciones es un elemento crítico (Leiras et al., 2014) [2].

La Logística Humanitaria es algo nuevo, pero ha crecido en términos de cantidad y relevancia en los últimos años. Además, como se menciona al comienzo de la Sección 1 de Introducción, es un tema de alto impacto en la sociedad, ya que si se mejoran los métodos utilizados en las operaciones humanitarias se pueden salvar vidas. Por estos motivos nuestra investigación tomará la dirección de la Logística Humanitaria centrándonos en los diferentes problemas de localización hasta el punto de profundizar en los métodos y conceptos que hay detrás del preposicionamiento de inventario

El resto del documento está organizado de la siguiente manera: En las siguientes Subsecciones de la Sección 1 de Introducción se hará hincapié sobre Logística Humanitaria, problemas de localización y particularmente sobre el preposicionamiento de inventario en Logística Humanitaria. En la Sección 2 de Definición del problema se describen los conceptos básicos para un mejor entendimiento del documento y se clasifican los diferentes problemas de localización, además se describe el problema de localización propuesto por Lee & Lee (2010) y se detallan las alternativas a dicho problema agregando otras variables de decisión como el costo de preposicionamiento y la capacidad de los centros de ayuda. En la Sección 3 de Algoritmos a desarrollar se describen los modelos y los algoritmos implementados. Como primer paso se describe el procedimiento heurístico propuesto por Lee & Lee (2010) y luego las modificaciones realizadas para nuestro problema. Por último, se presentan las optimizaciones al algoritmo propuesto en este trabajo para poder aumentar el cubrimiento, pero manteniendo la medida de costo-efectividad de la solución de partida. También se explica el funcionamiento del sistema implementado y de las distintas funcionalidades que posee como por ejemplo la generación y visualización de instancias, entre otras. En la Sección 4 de Diseño y en la Sección 5 de Implementación se describen el diseño y arquitectura del sistema y las tecnologías utilizadas para la implementación de los algoritmos respectivamente. En la Sección 6 de Testing se detalla la metodología empleada para la realización de las pruebas de evaluación de los algoritmos, así como de cada uno de los componentes. Estas pruebas se centran en la robustez del software resultante. En la Sección 7 de Resultados obtenidos se hace un análisis y comparación de las soluciones obtenidas en donde se presentan tablas con los resultados para cada tipo de problema. Por último, en la Sección 8 de Conclusiones y trabajo a futuro se hace una síntesis de lo estudiado y un análisis de las diferentes alternativas para resolver el problema de localización en Logística Humanitaria y cuales dieron mejor resultado en el contexto de este trabajo. Además, se comentan posibles trabajos a futuro para la continuidad de lo realizado.

1.1. Logística Humanitaria

La Logística Humanitaria es casi el 80% del total de los esfuerzos en el apoyo a desastres (Trunick, 2005) [5].

La actividad principal de la Logística Humanitaria consiste en planear, implementar y controlar eficientemente el costo del flujo del almacenamiento de materiales y bienes, al igual que toda información relacionada, desde el punto de origen hasta el punto de consumo, con el propósito de ayudar a las personas vulnerables (afectadas). Dado que la demanda es impredecible (Balcick & Beamon, 2008) [6], esta actividad no es sencilla.

Por lo tanto, Logística Humanitaria se puede definir como: el proceso de planificación, implementación y control de la eficiencia, flujo rentable y almacenamientos de bienes materiales, como también información relacionada, desde el punto de origen al punto de consumo para satisfacer los requerimientos de los beneficiarios (afectados, víctimas) (Leiras et al., 2014) [2].

Optimizar todo esto requiere que todo lo relacionado entre los actores sea administrado a través de un enfoque que eficientemente y eficazmente coordine las diferentes organizaciones, eliminando redundancia, maximizando la eficiencia a lo largo de la Cadena de Suministros. Por lo tanto la Cadena de Suministros se enfoca en la relación que hay entre los actores para que la movilización (la definición de Logística Humanitaria) sea posible. Chopra & Meindl (2008) [7] definen de la siguiente manera Cadena de Suministros (CS): Una Cadena de Suministros está formada por todas aquellas partes involucradas de manera directa o indirecta en la satisfacción de una solicitud de un cliente. La Cadena de Suministros se ha dado a conocer como la clave para el éxito en la Logística Humanitaria, sin embargo, no llama la atención de los investigadores. El objetivo de la Cadena de Suministros es proveer rápidamente apoyo a las áreas afectadas para minimizar el sufrimiento humano y la muerte (Chopra & Meindl ,2008) [7].

En la cadena de suministros, la efectividad nos asegura que lo haremos en el menor tiempo posible, ahorrar tiempo significa mayor cantidad de vidas salvadas. Mientras que la eficiencia nos asegura el tener un menor gasto monetario, lo que significa poder ayudar a una mayor cantidad de personas.

Akkiha (2006) [8] divide el proceso de la Logística Humanitaria en 3 escenarios: Preparación -enfocado antes de que el desastre se manifieste-, Respuesta -inmediatamente después de que el desastre se manifiesta-, y recuperación. Cuando el desastre se manifiesta, las primeras acciones a tomar son: solicitud de donaciones y fondos a los donadores. Los suministros son obtenidos a través de acuerdos preestablecidos con los vendedores, y son generalmente adquiridos durante la etapa de preposicionamiento. Los suministros recibidos de los donantes y

los suministros adquiridos de los vendedores son transportados por diversos medios a lugares predeterminados y distribuidos por los servicios de emergencia a las zonas afectadas. La complejidad de la Logística Humanitaria se aprecia cuando el proceso de distribución a través de la línea del tiempo se ve afectado por los factores y características de la cadena de suministros.

1.2. Actores

La ayuda humanitaria involucra muchos actores de diferentes índoles, con un alto grado de heterogeneidad (en términos de cultura, propósito, intereses, capacidad y experiencia logística). Los actores claves se pueden categorizar como los siguientes: los gobiernos, los militares, las agencias de ayuda, los donantes, organizaciones no gubernamentales (ONG) y empresas del sector privado (Cozzolino, 2007) [3].

Los gobiernos (gobierno local, gobierno de países vecinos, entre otros) son los que de alguna manera están encargados de activar el flujo de la Logística Humanitaria luego de un desastre, ya que son ellos los que tienen el poder para autorizar operaciones y movimiento de recursos. En efecto, sin la autorización del gobierno local, no sería posible que ninguna otra organización o ente de ayuda pueda participar. Esto es un tema delicado ya que puede haber conflicto entre los países locales y vecinos. Existen acuerdos que se firman previamente, el cual en caso de desastre, los países que la integran participan en la ayuda del país afectado (Cozzolino, 2007) [3].

1.3. Problemas de Localización en Logística

Una forma adecuada de abordar los problemas de apoyo humanitario relacionados al preposicionamiento de suministros, es considerándolos como problemas de localización, en donde los clientes son las víctimas afectadas por el desastre, los depósitos son los centros de ayuda en donde se almacenan suministros y se resguarda a los refugiados. Las personas afectadas a su vez tienen que ser satisfechas de ciertos bienes y servicios (primeros auxilios, comida, agua, etc.).

Por lo tanto se busca el lugar en dónde ubicar las instalaciones de suministros, tomando en cuenta diferentes requerimientos, con el objetivo de minimizar tanto los tiempos de respuesta ante el desastre cómo los precios de los suministros preestablecidos. Esto es muy importante ya que una característica de los desastres es la ocurrencia repentina de los mismos, por lo que se necesita una respuesta inmediata en las áreas devastadas.

En la siguiente Subsección, se detalla la técnica de preposicionamiento de inventario en Logística Humanitaria para resolver los problemas de localización.

1.4. Preposicionamiento de Inventario

El preposicionamiento de inventario en Logística Humanitaria consiste en distribuir y posicionar previamente de forma apropiada ciertos elementos que son de utilidad para dar apoyo a los afectados por desastres naturales.

El problema se puede traducir a un problema de localización (mencionados en la Sección 2 de Definición del problema), en donde los elementos de utilidad son almacenados en distintas instalaciones, y el problema consiste en dónde ubicar esas instalaciones con el fin de satisfacer la demanda. Para esto se deben tener en cuenta los requerimientos del problema en particular.

Akkiha (2006) [8] divide el preposicionamiento del inventario en dos categorías. La primera categoría consiste en la estimación de los suministros requeridos en los puntos de demanda a lo largo de la cadena de suministros, y en el reconocimiento de patrones por los cuales se disparan los eventos que hacen que se requieran suministros en el punto de demanda dado. La segunda categoría es la que examina los aspectos de espacio en las operaciones. Esta categoría explora la dinámica de la ubicación geográfica de las instalaciones con respecto a otros factores como el costo y los servicios. La diferencia entre las categorías es que la primera categoría refiere a qué cantidad de suministros se necesitarán en un punto de demanda dado, y la segunda categoría refiere a donde deben estar ubicados esos suministros para mejorar la calidad del servicio brindado.

Como se dijo en la Sección 1 de Introducción, el proceso de la logística humanitaria se divide en tres fases a lo largo del tiempo: esfuerzos antes de que el desastre ocurra, respuesta inmediata luego de que el desastre ocurre, y la recuperación post-desastre. El primer período es estratégico: El desastre aún no ha ocurrido pero el preposicionamiento de inventario y las preparaciones en cuanto a infraestructura ya han tomado lugar anticipando el desastre. El preposicionamiento de activos puede incluir la expansión de estos depósitos, instalaciones médicas, refugios temporales, mientras que la preparación de infraestructura puede incluir previsiones en las pistas de aterrizaje aéreas y espacios de rampa en las mismas (Apte, 2012) [9].

Ubicar de forma adecuada los depósitos juega un rol importante en la estrategia del preposicionamiento. La gran pregunta que surge es: ¿Qué activos deben estar ubicados en la anticipación a un desastre? y además, ¿Dónde deben estar ubicados? Luego de ubicar los

depósitos y de realizar el preposicionamiento de activos, el inventario de los suministros críticos debe ser bien administrado (Apte, 2012) [9].

La preparación trae consigo ciertos problemas, como ser donde ubicar los depósitos, el preposicionamiento de activos, ubicación de recursos, y planificar el transporte en anticipación a los desastres. La respuesta inmediata trae los problemas de manejo de inventario, distribución, toma de decisiones a lo largo de la crisis. Y los problemas de la recuperación post-desastre son transporte, evacuación, manejo del tráfico, y otros.

El sitio web de UNHRD (http://www.unhrd.org/) ofrece una lista de 122 ítems que recomienda almacenar en los depósitos. Algunos de los ítems recomendados son: Medicamentos y equipos médicos, dispositivos de electricidad, alimentos, kits individuales, radios y telecomunicaciones, elementos de seguridad, saneamiento e higiene, refugio y vivienda, entre otros. La entrega conveniente de estos recursos, son críticos para la fase inicial de respuesta, ya que permiten una rápida instalación de la infraestructura, mejorando la entrega de bienes y servicios a los necesitados (Akkiha ,2006) [8].

Dado que el preposicionamiento toma parte en la fase de preparación, en la siguiente Sección se entrará en detalle en esa fase.

1.4.1. Fase de preparación

La preparación en cuanto a anticipación a desastres involucra preposicionamiento de activos y ubicación de recursos. Una parte significante de ello involucra donde ubicar los depósitos y los centros de distribución. Por otro lado, también involucra la planificación para el posible reemplazo de vehículos. Además, como se mencionó anteriormente, involucra las respuestas a qué almacenar, donde y cuando.

1.4.2. Ubicación de Instalaciones

Encontrar una ubicación robusta para los centros de ayuda que no solamente va a ser útil en base a los requerimientos actuales y del estado del sistema, sino que seguirá siendo la mejor ubicación ante cambios en el sistema en términos de clima, densidad de población y tendencias de mercado puede no ser una preocupación inmediata durante una crisis, pero es relevante en el contexto de la Logística Humanitaria (Apte, 2012) [9].

El estudio del problema de preposicionamiento de suministros ha sido abordado usando mayormente dos tipos de estrategias: *Set Covering Problem* (SCP) y *Facility Location Problem* (FLP) como describimos con mayor profundidad en el Anexo 1 de Problemas de Localización.

Ambas estrategias deciden acerca de la ubicación de las instalaciones, pero con criterios de optimización diferentes.

2. Definición del problema

El problema de localización que se considerará en este trabajo se puede definir de la siguiente manera: encontrar la mejor ubicación para los centros de ayuda de tal forma que maximice la cobertura, minimizando la suma de los costos de instalación involucrados. Existe un conjunto de posibles ubicaciones para los centros de ayuda, un conjunto de ubicaciones de poblaciones (nodos) de personas afectadas, así como diferentes suministros requeridos y niveles para los centros de ayuda. Cada centro de ayuda posee un volumen máximo de capacidad para el almacenamiento de los suministros, y un costo asociado por posicionar una unidad de suministro en un centro de ayuda. Cada centro de ayuda posee diferentes niveles, en los cuales a mayor nivel, mayor variedad de suministros brindará, es decir un centro de nivel *k* podrá brindar los suministros de tipo 1 a *k*. A su vez, a mayor nivel, mayor será el costo de abrir el centro de ayuda. Para cada suministro brindado por un centro de ayuda habrá asociado un radio de cobertura total y un radio de cobertura parcial. El primero representará que todas las personas que estén ubicadas dentro de dicho radio podrán ser cubiertas totalmente por el suministro que requieren. El segundo (radio de cobertura parcial) determina que a medida que el nodo de personas afectadas se aleja del centro de ayuda, tendrá un valor de cobertura menor, hasta el punto en el que esté fuera del radio de cobertura parcial, en el cual el cliente ya no será cubierto por dicho centro de ayuda. Cada nodo de población tiene una cierta cantidad de personas que van a requerir un suministro *k* y está ubicado en una cierta distancia de cada centro de ayuda. El objetivo es determinar una asignación de poblaciones a centros de ayuda, de manera de maximizar el cubrimiento de la demanda de las personas afectadas por un desastre natural, minimizando los costos de preposicionamiento asociado, dichos costos son tanto el costo de instalación de los centros de ayuda a abrir como los costos asociados a la localización de suministros. Para lograr dicho objetivo se decidió hacer uso de la formulación del problema como un caso del *Hierarchical Covering Location Problem*, el problema se describe en Anexo 1 de Problemas de Localización.

El estudio se centró en *Hierarchical Covering Location Problem* (variante del Set Covering) antes que las otras variantes existentes para abordar problemas de localización, porque en los problemas en los que están en juego vidas humanas, no es lo más conveniente minimizar la suma de las distancias (p-median) ni minimizar la máxima distancia (p-center). Esto es porque la solución óptima de p-median puede ser que incluya a una persona muy alejada de su centro de ayuda más cercano, mientras que en p-center la solución óptima puede hacer que haya demasiadas personas igual de lejos que la máxima distancia, lo que puede no ser adecuado si

se tienen en cuenta criterios de eficiencia y equidad. Aplicando *Hierarchical Covering Location Problem* lo que se busca es que las personas estén ubicadas a no más de una distancia prefijada de su centro de ayuda más cercano, incluyendo tantos centros de ayuda como sea necesario para cumplir con el objetivo. Cumplir con el objetivo, requerirá entonces que se pueda contar con la cantidad de centros de ayuda necesarios para cubrir a todas las personas.

Para llegar al objetivo del trabajo, como primer punto se requiere implementar el algoritmo propuesto por Lee & Lee (2010). Luego, se debe diseñar una alternativa para el mismo, para incluir los costos de preposicionamiento y la capacidad de los centros de ayuda.

Cómo última parte, se requiere comparar las soluciones obtenidas entre los algoritmos implementados, así como también la comparación contra las soluciones obtenidas por el solver GLPK, tomando como medidas de comparación la cobertura obtenida, el costo obtenido, la relación costo-efectividad y el tiempo de ejecución.

El Costo-Efectividad (CE) determina el costo asociado por unidad de servicio brindado a cada persona afectada. Es de interés no sólo maximizar el cubrimiento, sino también minimizar el CE, ya que de esta manera se está brindando un servicio de forma más eficiente desde el punto de vista económico.

A continuación se presentan los componentes de la formulación matemática para el problema de localización para maximizar el cubrimiento propuesto por Lee & Lee (2010), el cual utiliza la técnica *Hierarchical Covering Location Problem.*

Parámetros:

- I*:* Cantidad total de potenciales ubicaciones para centros de ayuda.
- J: Cantidad total de ubicaciones donde hay personas que necesitan ser asistidas.
- K*:* Cantidad total de tipos diferentes de paquetes de suministros de ayuda. Se asume que un paquete de suministros de tipo i es más necesario que un paquete de tipo j, con $i < j$.
- L*:* Cantidad total de niveles de servicio diferentes de un centro de ayuda, con $L \geq K$. Un centro de ayuda con un nivel de servicio k podrá proveer paquetes de suministros de tipo $1, \dots, k$.
- c_{ijk}: Radio de cobertura que provee el centro de ayuda i a las personas afectadas en el lugar j para el paquete de suministros de tipo k, con $0 \leq c_{ijk} \leq 1, i = 1, \dots, I, j = 1, \dots, J, k = 1, \dots, K$.

- f_{jk}: Número total de personas de la población en el lugar j que requiere del suministro de tipo k, con $j = 1, \dots, J, k = 1, \dots, K$.
- p_l: Cantidad máxima de centros de ayuda con nivel l que pueden ser instalados, con $l = 1, \dots, L$.
- t_{ik}: Cantidad máxima de paquetes de suministros de tipo k que se puede preposicionar en el centro de ayuda i, con $i = 1, \dots, I, k = 1, \dots, K$.

Variables de decisión:

- x_{ijk}: 1 si la demanda de suministros del tipo k de la población ubicada en el lugar j es cubierta por el centro de ayuda instalado en el lugar i; 0 en caso contrario, con $i = 1, \dots, I, j = 1, \dots, J, k = 1, \dots, K$.
- y_{il}: 1 si un centro de ayuda de nivel l se instala en la ubicación i; 0 en caso contrario, con $i = 1, \dots, I, l = 1, \dots, L$.

Modelo:

$$\max_{x,y} \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} c_{ijk} \times f_{jk} \times x_{ijk} \quad (1)$$

Sujeto a:

$$x_{ijk} \leq \sum_{l=k}^{L} y_{il}, \ \forall i, j, k \quad (2)$$

$$\sum_{i \in I} x_{ijk} \leq 1, \ \forall j, k \quad (3)$$

$$\sum_{i \in I} y_{il} \leq p_l, \ \forall l \quad (4)$$

$$\sum_{j \in J} f_{jk} \times x_{ijk} \leq t_{ik}, \ \forall i, k \quad (5)$$

$$\sum_{l \in L} y_{il} \leq 1, \ \forall i \quad (6)$$

$$x_{ijk} = \{0,1\} \forall i, j, k \quad (7)$$

$$y_{il} = \{0,1\} \forall i, l \quad (8)$$

En (1) se plantea el objetivo de maximizar la cobertura de las poblaciones afectadas por el desastre, teniendo en cuenta el radio de cobertura parcial o total. Las restricción (2) indica que la demanda de un determinado tipo de suministro k de una población j puede ser cubierta por un centro de ayuda en el lugar i si y solamente si se instala un centro de ayuda en i con nivel l con $l \geq k$. Las restricciones (3) representan que la demanda de un suministro de tipo k de una

población j puede ser cubierta a lo sumo por un solo centro i. Las restricciones de (4) indican que a lo sumo se pueden instalar en total una cantidad p_l de centros con nivel l. En (5) se indica que la cantidad total de suministro tipo k que se distribuye desde el centro instalado i no puede superar la cantidad preposicionada en ese centro. Las restricciones (6) indican que en un lugar i solo puede ser instalado un único centro de ayuda sin importar el nivel del mismo. Las restricciones (7) y (8) establecen el dominio de valores posibles para las variables de decisión.

El radio de cobertura c_{ijk} se calcula de la siguiente manera (Lee & Lee, 2010): $c_{ijk} = f(R_{ij}, S_k, L_k)$ en donde R_{ij} es la distancia entre el centro de ayuda i y el nodo j de personas afectadas. S_k es la distancia para la cual se entiende que la cobertura es total para el tipo de paquete de suministros k, mientras que L_k es la distancia máxima para una cobertura parcial.

La función f se definirá de tal forma que se cumplan las siguientes condiciones para los valores de radio de cobertura:

- c_{ijk} = 1 si $R_{ij} \leq S_k$.
- $0 < c_{ijk} < 1$ si $S_k < R_{ij} < L_k$ (en este caso el c_{ijk} será el cociente entre $\frac{R_{ij}}{L_k}$)
- c_{ijk} = 0 si $R_{ij} \geq L_k$

El modelo presentado puede tener un impacto negativo en los costos tanto de preposicionamiento como de instalación de los centros de ayuda, ya que no los considera. Teniendo en cuenta además que desde el punto de vista humanitario se le quiere dar más importancia al cubrimiento que a los costos, se decidió incluir como un problema subordinando al de máximo cubrimiento, el problema de minimizar los costos de instalación de centros de ayuda y de preposicionamiento de suministros. De esta manera la solución óptima será aquella que maximiza el cubrimiento, pero al costo mínimo de localización y preposicionamiento. Por lo tanto se modelará el problema con programación Bi-Nivel el cual se puede definir de la siguiente forma (Talbi, 2013) [20]:

$\min_{x \in X} F(x, y)$

Sujeto a

$G(x, y) \leq 0$
$min_{y \in Y} f(x, y)$
Sujeto a: $g(x, y) \leq 0$
$x, y \leq 0$
$x \in \mathbb{R}^{n_1}$, $y \in \mathbb{R}^{n_2}$.

Este tipo de programación expresa una relación de jerarquía entre problemas, lo cual es la característica principal de los problemas Bi-Nivel. Por lo tanto, se le denomina a la parte superior

función objetivo de nivel superior, o problema de primer nivel, y a la parte inferior se le denomina problema de nivel inferior.

La jerarquía expresada por este tipo de programación, es útil en muchos contextos en donde se quiere llegar a una solución óptima tanto del problema de nivel superior, cómo del problema de nivel inferior, pero indicando una preferencia por el problema de primer nivel. En el contexto de la Logística Humanitaria, y aplicando la técnica *Hierarchical Covering Location Problem,* el problema de nivel superior es satisfacer la demanda de las personas afectadas de la mejor manera posible (maximizar la cobertura), mientras que el problema de nivel inferior será minimizar costos.

3. Algoritmos a desarrollar

En esta Sección se detalla funcionalmente lo realizado para cumplir con los requerimientos definidos en la Sección 2 de Definición del problema. En dónde como primer paso, se desarrolló la heurística propuesta por Lee & Lee (2010) para tener una referencia de base. Luego, se implementó una variante al problema base, en dónde se tienen en cuenta costos, y se realiza una formulación Bi-Nivel. Por último, y una vez obtenidas las formulaciones comentadas previamente, se implementaron optimizaciones para poder aumentar el cubrimiento obtenido tratando de minimizar el costo efectividad.

Las soluciones obtenidas por la resolución del modelo Bi-Nivel serán iguales o de peor calidad que las obtenidas por la resolución del modelo propuesto por Lee & Lee (2010) en cuanto a cobertura obtenida. Esto se debe a que el modelo Bi-Nivel cuenta con un espacio de soluciones más acotado, ya que no solo se quiere maximizar la cobertura obtenida, sino que también se quiere minimizar el costo efectividad. Tampoco es posible obtener la solución óptima del modelo Bi-Nivel mediante el software de resolución utilizado debido a las características del modelo, por ende no se sabe qué tan alejadas estarán las soluciones del algoritmo implementado para resolver el problema Bi-Nivel. Sin embargo, se sabe que dada una instancia, la mejor solución obtenida mediante el modelo Bi-Nivel nunca será mejor que una solución obtenida mediante el modelo propuesto por Lee & Lee (2010) en cuanto a cobertura obtenida. La diferencia de cobertura entre la solución obtenida por el algoritmo propuesto para resolver el problema Bi-Nivel, y el obtenido por el uso del software para resolución del modelo Lee & Lee (2010) puede ser muy grande, por lo tanto y a modo de poder medir la calidad de la solución obtenida, se diseñó un modelo matemático que utilizando el costo obtenido por la solución como cota máxima, busca maximizar la cobertura. Dicho modelo se explica en la Sección 3.2.5 de Problema con costos acotados.

Al final de esta Sección se explica a grandes rasgos el funcionamiento del sistema implementado y de las distintas funcionalidades que posee como por ejemplo la generación y visualización de instancias, entre otras.

3.1. Algoritmo de Lee & Lee

Se implementó el algoritmo propuesto por Lee & Lee (2010) para poder comparar los diferentes algoritmos propuestos para el problema Bi-Nivel. A continuación se describe un pseudocógido del procedimiento heurístico de Lee & Lee (2010) para el problema de maximizar

la cobertura conocido en la literatura como *Hierarchical Covering Location Problem (HLCP)*. La especificación del modelo matemático del problema está en la Sección 2 de Definición del problema. Cabe aclarar que para el problema estudiado por Lee & Lee (2010) no se tiene en cuenta ningún costo asociado a la localización ni preposicionamiento de suministros.

La heurística consta de dos partes, la primera consiste en encontrar una solución inicial a través de la técnica Greedy, y la segunda parte consiste en tomar la solución inicial y mejorarla a través de la metaheurística Tabú Search.

3.1.1. Solución inicial

La primera parte de la heurística consiste en obtener una solución inicial factible utilizando la técnica de Greedy. Por lo tanto, la solución inicial se construye seleccionando las ubicaciones de los centros de ayuda a partir de un conjunto de posibles ubicaciones, luego se le asignan los nodos de personas afectadas. Cada centro de ayuda tiene asociado un valor de cobertura que es el máximo que podría brindar, este valor es la suma de todas las necesidades de las personas afectadas que están dentro del radio de cobertura del centro. En la solución inicial se seleccionan los centros de ayuda con un mayor valor de cobertura asociado (técnica Greedy) hasta alcanzar el valor de centros de ayuda máximo por nivel (parámetro P_l). La selección de centros de ayuda se realiza empezando por los niveles más bajos a los más altos.

La asignación de los nodos de personas afectadas también se realiza de una manera Greedy, asignando nodos de personas afectadas a los centros de ayuda que maximicen el total de la cobertura.

Notaciones:

TC: denota la cobertura total que se espera si el centro de ayuda está abierto.

OC: Denota la cobertura de cierto nodo de personas afectadas a cada centro de ayuda específico que está abierto.

1. Paso 1: selección de centro de ayuda a abrir

   1.1 Calcular la cobertura total

   $$TC_i = \sum_j \sum_k c_{ijk} f_{jk} \quad \forall i$$

   1.2 Determinar el conjunto de cobertura con aquellos centros que tengan un valor de cobertura positivo:

   $$I = \{i | TC_i > 0, \forall i\}$$

   1.3 Seleccionar el centro de ayuda a abrir:

1.3.1 $l = 1$

1.3.2 Inicializar en vacío el conjunto de centro de ayuda seleccionadas en el nivel l

$$I_l^* = \emptyset, \forall l$$

1.3.3 Seleccionar el centro de ayuda con mayor *TC,* y removerlo del conjunto I

$$Max_{i \in I}(TC_i)$$

1.3.4 Agregar i en el conjunto I_l^*

1.3.5 Si $l < |L| \wedge |I_l^*| \geq p_l$ entonces $l = l + 1$ e ir al punt 1.3.3

Si no, Si $l = |L| \wedge |I_l^*| \geq p_l$ entonces ir al paso 2

Si no ir al paso 1.3.3

2. Paso 2: Asignación de nodos de personas afectadas.

2.1 Calcular la cobertura de las instalaciones abiertas

$$OC_{ijl} = \sum_k c_{ijk} f_{jk}, \forall j, l, i \in I_l^*$$

2.2 Asignar los nodos de personas afectadas a los centros de ayuda

2.2.1Inicializar como vacío al conjunto de nodos de personas afectadas asignados a los centros de ayuda de nivel l

$$J_{il}^* = \emptyset, \forall i, l$$

2.2.2Construir el conjunto de nodos de personas afectadas cubiertos por el centro de ayuda i de nivel l

$$J_{il} = \{j | OC_{ijl} > 0, \forall j\}, \forall i, l$$

2.2.3Seleccionar el cliente $j = Max_{i \in I_l^*}(OC_{ijl})$, remover el nodo de personas afectadas seleccionado j del conjunto J_{il}

2.2.4Si $\sum_{j \in J_{il}^*} f_{jk} \leq t_{ik}$ entonces asignar el nodo de personas afectadas j al centro de ayuda tal que $OC_{ijl} = Max_{i^* \in I_{il}^*, j}(OC_{ijl})$. Agregar el nodo de personas afectadas j del conjunto J_{il}^*

Si no ir al punto 2.2.3

3.1.2. Procedimiento de mejora

En cada iteración del procedimiento de mejora se busca la mejor solución local a partir de las soluciones vecinas de la solución actual. Para generar esas soluciones vecinas se utiliza una lista tabú de manera de evitar la repetición de soluciones ya exploradas. Cada solución vecina se construye cerrando uno de los centros de ayuda abiertos en un nivel dado, y abriendo uno de

los centros de ayuda cerrado en el mismo nivel. Para el nuevo centro de ayuda abierto, se determina la asignación de los nodos de personas afectadas a dicho centro. El procedimiento de mejora se repite hasta que se alcanza el criterio de parada, el cual consiste en que la cantidad de iteraciones supere un valor pre definido

La selección del centro de ayuda a cerrar y el centro de ayuda a abrir se basa en la cobertura obtenida por los nodos de personas afectadas. Entre los centros de ayuda abiertos, aquel con menor cobertura asignada a nodos será candidato a ser cerrado. La cobertura asignada a un centro de ayuda i está dado por $AC_i = \sum_{j \in J_{il}^*} \sum_k c_{ijk} f_{jk}$. Él AC se diferencia del TC o del OC en que tanto TC como OC son coberturas esperadas, mientras que el AC es la cobertura actual que tiene el procedimiento de mejora hasta el momento. El centro de ayuda candidato a abrir será aquel que tenga mayor TC. La asignación de los nodos de personas afectadas a centros de ayuda es realizada de manera Greedy.

En el procedimiento iterativo, la lista tabú para el nivel l indica la lista de los centros de ayuda recientemente cerrados en ese nivel. La lista tabú está implementada de la siguiente manera: n listas, en donde n es la cantidad de niveles del problema, cada nivel posee una lista de elementos, en donde cada elemento es un centro de ayuda. La lista tabú es actualizada agregando el centro de ayuda con el menor AC, quitando el primer centro de ayuda asignado a la lista para un cierto nivel. Inicialmente el tamaño de la lista tabú de cada nivel es de tres, el tamaño puede incrementar si en 500 iteraciones no se encuentra una mejor solución. Existe una cota para el tamaño de la lista, por cada nivel, su correspondiente lista no puede superar el valor de $(m_l - p_l - 1)$ donde m_l es el número de potenciales ubicaciónes para los centros de ayuda con nivel *l*, y p_l es la cantidad de instalaciones permitidas a abrir en el nivel l. El procedimiento de selección de potenciales ubicaciones para los centros de ayuda como también la asignación de nodos de personas afectadas a dichos centros es la siguiente:

1. Paso 1: Intercambio de los centros de ayuda por nivel
    1.1 Calcular la cobertura de asignación
    $AC_i = \sum_{j \in J_{il}^*} \sum_k c_{ijk} f_{jk} \quad \forall i \in I_l^*$
    1.2 Seleccionar el centro de ayuda a cerrar
        1.2.1 Seleccionar el centro de ayuda abierto con $Min_{i \in I_l^*}(AC_i)$
        1.2.2 Remover el centro de ayuda seleccionado del conjunto I_l^*
        1.2.3 Actualizar la lista tabú
        1.2.4 Seleccionar el centro de ayuda cerrado con $Max_{i \in I_l}(TC_i)$

1.2.5 Remover el centro de ayuda del conjunto I

1.2.6 Agregar el centro de ayuda a abrir al conjunto I_l^*

2. Paso 2: Asignación de nodos de personas afectadas.

2.1 Calcular la cobertura de las instalaciones abiertas

$OC_{ijl} = \sum_k c_{ijk} f_{jk}, \forall j, l, i \in I_l^*$

2.2 Asignar nodos de personas afectadas a centros de ayuda

2.2.1 Inicializar como vacío al conjunto de nodos de personas afectadas asignados a los centros de ayuda de nivel l

$J_{il}^* = \emptyset, \forall i, l$

2.2.2 Hacer el conjunto de nodos de personas afectadas cubiertos por el centro de ayuda i de nivel l

$$J_{il} = \{j | OC_{ijl} > 0, \forall j\}, \forall i, l$$

2.2.3 Seleccionar el cliente $j = Max_{i \in I_l^*}(OC_{ijl})$, remover el nodo de personas afectadas seleccionado j del conjunto J_{il}

2.2.4 Si $\sum_{j \in J_{il}^*} f_{jk} \leq t_{ik}$ entonces asignar el nodo de personas afectadas j al centro de ayuda tal que $OC_{ijl} = Max_{i^* \in I_{il}^*, j}(OC_{ijl})$. Agregar el nodo de personas afectadas j del conjunto J_{il}^*

Si no ir al punto 2.2.3

2.2.5 Si $|J_{il}| = 0, \forall i, l$ entonces ir al siguiente paso

Si no ir a 2.2.4

El procedimiento de mejora termina cuando se cumple el criterio de terminación, dicho criterio es la cantidad de iteraciones del proceso, siendo un total de 3000 iteraciones.

3.2. Formulación y algoritmo para el problema Bi-Nivel

Para el modelo a construir se tomará como base el modelo propuesto por Lee & Lee (2010), agregando como una de sus restricciones un modelo para minimizar los costos de instalación y de preposicionamiento, de esta manera se obtiene una formulación Bi-Nivel, de forma similar al modelo multi-nivel de Irohara et al. (2013). De esta manera, se puede considerar que la formulación obtenida es una extensión de la realizada por Lee & Lee (2010) para el *Hierarchical Covering Location Problem*.

Dado que el objetivo principal es maximizar el cubrimiento de los centros de ayuda, nuestra función objetivo seguirá siendo la misma que en el modelo propuesto por Lee & Lee (2010). Sin

embargo, se modificarán las restricciones del problema, construyendo así un problema Bi-Nivel en donde se tienen en cuenta los costos asociados a los centros de ayuda y al preposicionamiento de suministros. El problema de primer nivel será el de maximizar el cubrimiento de los centros de ayuda a las poblaciones afectadas por el desastre, mientras que el problema de segundo nivel tendrá foco en minimizar los costos asociados al preposicionamiento de suministros y de la apertura de los centros de ayuda. Además, se introducen otros conceptos como los de volumen que ocupa cada elemento de un tipo de suministro, y capacidad total de un centro de ayuda.

El enfoque Bi-Nivel permite trabajar con una preferencia entre los problemas, permitiendo dar más importancia a los problemas de mayor nivel, haciendo que los niveles inferiores reaccionen con respecto a las decisiones tomadas en los niveles superiores. En nuestro caso lo más importante son las vidas humanas, por lo que la función objetivo del primer nivel es maximizar de la cobertura, dejando como segundo nivel el problema asociado a la minimización de los costos.

El modelo propuesto es el que se brinda a continuación. Notar que solo se detallan los parámetros, variables de decisión, y restricciones que se agregan al modelo propuesto por Lee & Lee (2010).

Parámetros:

- m_{ik}: Costo de preposicionar por unidad de suministro de tipo k en el centro de ayuda instalado en i, con $i = 1, \dots, I, k = 1, \dots, K$.
- S_{il}: Costo fijo de instalar un centro de ayuda de tipo l, en el lugar i, con $i = 1, \dots, I, l = 1, \dots, L$.
- α_k: Fracción de la demanda que se debe cubrir como mínimo para el tipo de suministros k, con $k = 1, \dots, K$.
- *M*: Valor entero grande, al menos igual a la suma de todas las demandas.
- v_k: Volumen de un paquete de suministros de tipo k, con $k = 1, \dots, K$.
- E_i: Volumen máximo del centro de ayuda instalado en el lugar i, con $i = 1, \dots, I$.

Variables de decisión:

- t_{ik}: Cantidad máxima de paquetes de suministros de tipo k que se van a preposicionar en el centro de ayuda i, con $i = 1, \dots, I, k = 1, \dots, K$.

Modelo

$$\max_{x,y} \sum_{i\in I} \sum_{j\in J} \sum_{k\in K} c_{ijk} \times f_{jk} \times x_{ijk} \quad (9)$$

Sujeto a:

$$x_{ijk} \leq \sum_{l=k}^{L} y_{il}\,,\ \forall i,j,k \quad (10)$$

$$\sum_{i\in I} x_{ijk} \leq 1,\ \forall j,k \quad (11)$$

$$\sum_{i\in I} y_{il} \leq p_l,\ \forall l \quad (12)$$

$$\sum_{j\in J} f_{jk} \times x_{ijk} \leq t_{ik},\ \forall i,k \quad (13)$$

$$\sum_{l\in L} y_{il} \leq 1,\ \forall i \quad (14)$$

$$x_{ijk} = \{0,1\} \forall i,j,k \quad (15)$$

$$y_{il} = \{0,1\} \forall i,l \quad (16)$$

$$\min_{y,t} \sum_{i\in I} \sum_{l\in L} S_{il} y_{il} + \sum_{i\in I} \sum_{k\in K} m_{ik} t_{ik} \quad (17)$$

Sujeto a:

$$\sum_{l\in L} y_{il} \leq 1,\ \forall i \quad (18)$$

$$t_{ik} \leq M\left(\sum_{l=k}^{L} y_{il}\right),\ \forall i,k \quad (19)$$

$$\sum_{i\in I} t_{ik} \geq \sum_{j\in J} \alpha_k f_{jk}\,, \forall k \quad (20)$$

$$\sum_{k\in K} v_k t_{ik} \leq \min(E_i, \sum_{j\in J} \sum_{k\in K \wedge c_{ijk} > 0} v_k f_{jk}), \forall i \quad (21)$$

$$t_{ik} \geq 0\,, \forall k \quad (22)$$

$$y_{il} = \{0,1\}, \forall i,l \quad (23)$$

Las restricciones desde (9) hasta (16) inclusive fueron explicadas en la Sección 3.1 de Algoritmo de Lee & Lee.

La función objetivo (17) plantea la minimización de la suma de los costos de instalación de centros de ayuda y de preposicionamiento de suministros en los mismos. Las restricciones (18) son equivalentes a las restricciones de (14) pero para el problema de segundo nivel. En (19) se establece que solo se puede preposicionar un suministro de tipo k en el centro i si y solo si se va a instalar un centro de ayuda en el lugar i. Las restricciones de (20) indican que para cada suministro del tipo k se debe preposicionar en total al menos una cantidad que permita cumplir con la exigencia mínima establecida según el factor α_k y la cantidad de habitantes f_{jk} que requieren ese suministro en cada población j. En (21) se establece que el volumen de todos los suministros preposicionados en un centro de ayuda instalado en i no puede superar el mínimo

entre la capacidad de dicho centro y el máximo cubrimiento posible que puede tener dicho centro. Finalmente, las restricciones (22) y (23) indican el dominio de valores posibles para las variables de decisión del segundo nivel.

Para el radio de cobertura c_{ijk} se utiliza el cálculo detallado en la Sección 3.1 de Algoritmo de Lee & Lee.

A lo largo del estudio del problema, el modelo matemático fue evolucionando para mejorar las soluciones obtenidas. El cambio más notorio corresponde a la restricción (21), donde inicialmente se tenía: $\sum_{k \in K} v_k t_{ik} \leq E_i \; \forall i$ esta restricción no se adapta de la mejor manera a la realidad ya que permite que la cantidad preposicionada en un centro de ayuda supere lo máximo que podría brindarle a todos los nodos de personas afectadas que estén a su alcance. Esto se puede entender fácilmente si se supone el caso de un centro aislado, en donde no existan personas afectadas en el radio del centro, se puede ver que no tendría sentido preposicionar suministros en el mismo ya que no se podría brindar ayuda a las personas afectadas. Realizando las modificaciones en esta restricción se puede tener un mayor control sobre la cantidad preposicionada en cada centro de ayuda, evitando excesos.

3.2.1. Heurística para el problema bi-nivel

El modelo planteado (9) – (23) corresponde a una formulación de un problema Bi-Nivel. Estos problemas son generalmente no convexos y no diferenciables. Incluso en Jeroslow (1985) [19] se demostró que en la situación más sencilla de un problema de este tipo cuando el problema de segundo nivel es lineal, el problema es NP-Hard. Además, se puede considerar que el problema formulado en (9) – (23) es una extensión del *Hierarchical Covering Location Problem* (HCLP) el cual también es un problema NP-hard (Lee & Lee, 2010). Por lo tanto, se decidió la implementación de un procedimiento heurístico para resolver el problema. El mismo está basado en una extensión del procedimiento propuesto por Lee & Lee (2010) para el HCLP.

El algoritmo utilizado para resolver el problema se divide en dos fases: La primer fase consiste en obtener una solución inicial por medio del método de asignación jerárquico utilizando la técnica de Greedy, mientras que en la segunda fase se mejora la solución inicial, utilizando métodos heurísticos de ubicación y asignación basándose en la técnica de Tabu Search (Lee & Lee, 2010).

Se utiliza la metaheurística Tabu Search dado que se quiere explorar un gran rango del espacio de soluciones. A su vez, el procedimiento a utilizar es conocido por alcanzar una buena solución en un corto tiempo de procesamiento (Lee & Lee, 2010). En las heurísticas basadas en

Tabu Search, lo primero que se hace es construir una solución inicial, y a partir de ella obtener soluciones vecinas hasta llegar a una solución de buena calidad.

A continuación se describe cada una de las fases de la heurística a implementar.

Procedimiento de solución inicial

Para obtener la solución inicial del problema, se parte del mismo método que el propuesto para obtener una solución inicial en la heurística propuesta por Lee & Lee, sin embargo como en el modelo propuesto en el contexto de éste trabajo el parámetro t_{ik} dejó de ser un parámetro para pasar a ser una variable, se calcula cómo el máximo posible que precisa cada centro de ayuda para poder servir a todas las personas en cada tipo de suministro, utilizando como tope para la variable t_{ik} únicamente el volumen de cada centro de ayuda. En el algoritmo implementado, se puede partir tanto de una solución factible como de una que no lo sea.

Procedimiento de decisión de cambio de centros de ayuda:

Este procedimiento es idéntico al de Lee & Lee dado que lo que interesa en esta etapa es probar con la mayor cantidad de centros posibles.

Procedimiento de mejora

El procedimiento de mejora al igual que en el algoritmo propuesto por Lee & Lee (2010), se basa en dos etapas. La primera etapa se comporta de la misma manera que en el algoritmo que proponen los autores, por más información dirigirse a la Sección 4.1.2 de Procedimiento de Mejora.

En la segunda etapa es dónde se ven reflejados los cambios dado que t_{ik} dejó de ser un parámetro para ser una variable del problema Bi-Nivel. En esta etapa la variable t_{ik} en vez de ser dada, se calculará en base a costos, capacidad de los centros, volumen de los elementos de cada tipo de servicio, y en base a la cobertura de la demanda deseada. Es por esto que como parte de las restricciones del problema principal que proponen los autores Lee & Lee (2010), se agrega una restricción que sea un problema de segundo nivel que sea capaz de obtener los valores para las variables t_{ik} que minimicen costos, a una demanda apropiada en el contexto del problema. Para resolver el problema de segundo nivel y así obtener los valores de t_{ik}, se utilizó el solver GLPK dado que el problema es de Programación Lineal y es de esperar que se pueda resolver en un tiempo razonable. El problema de segundo nivel contiene la información de los centros abiertos, por lo tanto solamente se pretende asignar los suministros que satisfagan la demanda y que además la suma de los costos asociados al preposicionamiento de suministros sea la mínima posible.

A continuación se describirá el procedimiento para la mejora de la asignación de nodos de personas a centros de ayuda. Notar que desde el punto 2 en adelante se utiliza el mismo comportamiento que en la fase 2 del procedimiento de mejora propuesto por Lee & Lee (2010).

1. Dados los centros abiertos obtenidos por el procedimiento que resuelve el subproblema de primer nivel, resolver el problema de segundo nivel mediante GLPK para obtener los valores de t_{ik}.
2. Calcular la cobertura de las instalaciones abiertas definidas en el conjunto I_l^* de la primer etapa

$$OC_{ijl} = \sum_k c_{ijk} f_{jk}\,, \forall j, l, i \in I_l^*$$

3. Asignar nodos de personas a los centros de ayuda que fueron abiertos
   3.1 Inicializar como vacío al conjunto de nodos de personas afectadas asignadas a los centros de ayuda de nivel l: $J_{il}^* = \emptyset, \forall i, l$
   3.2 Construir el conjunto de nodos de personas afectadas cubiertos por el centro de ayuda i de nivel l

$$J_{il} = \{j | OC_{ijl} > 0, \forall j\}, \forall i, l$$

   3.3 Seleccionar el nodo $j = Max_{i \in I_l^*}(OC_{ijl})$, remover el nodo seleccionado j del conjunto J_{il}
   3.4 Si $\sum_{j \in J_{il}^*} f_{jk} + f_{j'k} \leq t_{ik}$ entonces asignar el nodo de personas afectadas j' al centro de ayuda tal que $OC_{ij'l} = Max_{i^* \in I_{il}^*, j}(OC_{ijl})$. Agregar el nodo de personas afectadas j' del conjunto J_{il}^*
   Si no ir al punto 2.3
   3.5 Si $|J_{il}| = 0, \forall i, l$ entonces ir al siguiente paso
   Si no ir a 2.4

Luego de finalizada cada iteración (incluyendo las etapas de decisión de cambio de centros de ayuda y la de asignación de personas afectadas), el procedimiento de mejora se repite hasta que se cumple el criterio de terminación, el cual se cumple cuando la cantidad de iteraciones alcance la cifra de 3000.

Con el algoritmo para resolver el problema Bi-Nivel, se logra obtener un cubrimiento que cumple con la demanda mínima aceptada de las personas afectadas, haciendo el costo lo más bajo posible. Para comparar la solución obtenida por el algoritmo para problema Bi-Nivel con la solución obtenida por el algoritmo propuesto por Lee & Lee (2010), se utilizó el concepto de Costo Efectividad como se explicó previamente. Con dicho concepto se llegó a que las soluciones

obtenidas por Bi-Nivel presentaban un valor de Costo Efectividad mejor que el obtenido por el algoritmo propuesto por Lee & Lee. Se entrará en más detalle de los resultados en la Sección 7 de Resultados Obtenidos.

Dado que el algoritmo Bi-Nivel obtiene soluciones con un buen Costo Efectividad, lo que se busca es aumentar el cubrimiento obtenido pero sin empeorar (o empeorando muy poco) el valor del Costo Efectividad. Con ese objetivo se realizaron dos optimizaciones tomando distintos enfoques que buscan una solución local a partir de la solución obtenida en la heurística propuesta para conseguir aumentar el cubrimiento sin que el Costo Efectividad se vea demasiado alterado. A continuación se describen las dos optimizaciones creadas para mejorar la solución obtenida a partir del algoritmo Bi-Nivel.

3.2.2. Optimización 1: Mejora a la solución obtenida por el algoritmo para el problema Bi-Nivel

Esta optimización busca una solución local a la solución obtenida en el algoritmo para el problema Bi-Nivel. La optimización 1 consta de dos partes, en dónde la primera pretende abrir la mayor cantidad de centros de ayuda que aún no estén abiertos a modo de mejorar la cobertura, y la segunda parte consiste en asignar nodos de personas afectadas a centros de ayuda abiertos. El resultado de la optimización cumple con todas las restricciones del modelo Bi-Nivel. Ambas partes son detalladas a continuación.

La primera parte de la optimización busca abrir centros de ayuda que estén cerrados siempre y cuando no se supere el límite máximo de centros de ayuda abiertos (parámetro P_l del modelo matemático). Para eso se recorren todos los centros de ayuda cerrados y se abre aquel centro que tenga mayor costo efectividad al abrirlo. Para determinar cuál es el centro de ayuda con mayor costo efectividad lo que se haces es, por cada centro cerrado se busca asignar todos los nodos de personas afectadas posibles. La asignación de nodos de personas afectadas se basa en asignar al centro de ayuda actual todos los nodos de personas afectadas que le den el mayor costo efectividad a dicho centro de ayuda. Una vez asignados todos los nodos de personas afectadas al centro de ayuda actual, se almacena temporalmente el valor del costo efectividad del centro de ayuda y todos los nodos de personas afectadas asignados, luego se elimina la asignación para iterar en el próximo centro de ayuda cerrado, repitiendo el procedimiento.

Una vez finalizada la primera parte de la optimización, con la solución resultado de la primera parte, se prosigue a la segunda parte de la optimización. Recorriendo todos los centros de ayuda

abiertos, y por cada centro de ayuda abierto se asigna todos los nodos de personas afectadas que le den el mayor costo efectividad a dicho centro de ayuda.

A continuación se describe en dos pseudocódigos lo mencionado anteriormente.

Este algoritmo se encargará de abrir los centros de ayuda que no hayan sido abiertos para una solución dada, solamente en caso de que la restricción P_l sea mayor a la cantidad de centros abiertos por nivel.

Se ordenará por Costo-Efectividad de la siguiente forma: $\frac{Q_i}{C_i}$ donde Q_i es el cubrimiento obtenido por el centro en cuestión, y C_i es el costo de abrir dicho centro y el costo de cubrir a las personas que tenga cubiertas. Al valor Costo-Efectividad del centro i le se le llamará CE_i

Entradas del algoritmo:

$SoluciónBiNivel$: La solución obtenida luego de ejecutar el algoritmo Bi-Nivel

$Problema$: Instancia del problema que se utilizó para resolver el algoritmo Bi-Nivel, el mismo contiene todas las variables y parámetros definidos en el modelo matemático

Variables:

I^*: Instalación a abrir

J^*: Nodos de personas afectadas a asignar

$CEactual$: Costo efectividad actual

I^C: Instalaciones cerradas

$Nivel$: Valor del nivel actual

Funciones:

$InstalacionesCerradas(Problema)$: Dado un problema/instancia retorna todos los centros de ayuda cerrados, ordenado de forma ascendente por costo efectividad

$ClientesNoServidos(Problema, Centro)$: Dado un problema/instancia y un centro de ayuda, retorna todos los nodos de personas afectadas que requieran un servicio y no se les haya asignado ningún centro de ayuda y el centro de ayuda pasada como parámetro puede satisfacerlos.

$CEAsignarClienteACentro(Centro, Clientes)$: Dado un centro de ayuda y una lista de nodos de personas afectadas, retorna el costo efectividad de asignar los nodos de personas afectadas al centro de ayuda pasado como parámetro.

$AsignarClientes(Solución, Centro, Clientes)$: Dado una solución, un centro de ayuda y una lista de nodos de personas afectadas, asigna los nodos de personas afectadas al centro de ayuda pasado como parámetro, guardándolo en la solución pasada como parámetro.

Pseudocódigo:

1. Optimización1A(SoluciónBi-Nivel, Problema)
2. $MejorSolución = SoluciónBiNivel$
3. $Nivel = 0$
4. **Mientras** $Nivel < Problema.L$ **hacer**
5. **Si** $Problema.pl > 0$ **entonces**
6. $I^* = null$
7. $J^* = nul$
8. $CEactual = \infty$
9. $I^C = InstalacionesCerradas(Problema)$
10. **Para cada** $i \in I^C$ **hacer**
11. $J^C = ClientesNoServidos(Problema, i)$
12. **Si** $CEactual > CEAsignarClienteACentro(i, J^C)$ **entonces**
13. $I^* = i$
14. $J^* = J^C$
15. $CEactual = CEAsignarClienteACentro(i, J^C)$
16. **Fin Si**
17. **Fin Para**
18. $AsignarClientes(MejorSolución, I^*, J^*)$
19. $Problema.pl = Problema.pl - 1$
20. **Si no**
21. $Nivel = Nivel + 1$
22. **Fin SI**
23. **Fin Mientras**
24. **Retornar** $MejorSolución$
25. FIN.

Algoritmo para maximizar la asignación de nodos de personas afectadas en centros de ayuda

Este algoritmo se encargará de asignar aquellos nodos de personas afectadas que no estén asignado a ningún centro de ayuda para un servicio que requieran. El algoritmo recorrerá todos los nodos de personas afectadas que no estén servidos y buscara la mejor asignación a un centro de ayuda intentando de tener el mejor CE al momento de asignarlo.

Entradas del algoritmo:

$SoluciónBiNivel$: La solución obtenida luego de ejecutar el algoritmo Bi-Nivel

$Problema$: Instancia del problema que se utilizó para resolver el algoritmo Bi-Nivel, el mismo contiene todas las variables definidas en el modelo matemático

Variables:

I^*: Centro de ayuda a abrir

J^*: Nodos de personas afectadas a asignar

$CEactual$: Valor de costo efectividad actual

Funciones:

$ClientesSinAsignacion(Problema)$: Dado un problema/instancia, retorna todos los nodos de personas afectadas que no tienen asignado un centro de ayuda para algún servicio.

$CentroCubreCliente(Centro, Cliente)$: Dado un centro de ayuda y un nodo de personas afectadas, determina si el centro de ayuda puede cubrir al cliente y además no sobrepasa el volumen del centro de ayuda.

$CEAsignarClienteACentro(Centro, Cliente)$: Dado un centro de ayuda y una lista de nodos de personas afectadas, retorna el costo efectividad de asignar los nodos de personas afectadas al centro de ayuda pasado como parámetro.

$AsignarClientes(Solución, Centro, Cliente)$: Dado una solución, un centro de ayuda y un nodo de personas afectadas, asigna el nodo de personas afectadas al centro de ayuda pasado como parámetro, guardándolo en la solución pasada como parámetro.

Pseudocódigo:

1. Optimización1B(SoluciónBi-Nivel, Problema)
2. $MejorSolución = SoluciónBiNivel$
3. $J^S = ClientesSinAsignacion(Problema)$
4. **Para cada** $j \in J^S$ **hacer**
5. $I^* = null$
6. $J^* = null$
7. $CEactual = \infty$
8. **Para cada** $i \in Problema.I$ **hacer**
9. **Si** $CentroCubreCliente(i, j)$ **entonces**
10. **Si** $CEactual > CEAsignarClienteACentro(i, j)$ **entonces**
11. $I^* = i$
12. $J^* = j$
13. $CEactual = CEAsignarClienteACentro(i, j)$
14. **Fin Si**
15. **Fin Si**
16. **Fin Para**
17. $AsignarClientes(MejorSolución, I^*, J^*)$
18. **Fin Para**
19. **Retornar** $MejorSolución$
20. FIN.

3.2.3. Optimización 2: Transformación de solución obtenida por el algoritmo para el problema Bi-Nivel basándose en la obtenida por el algoritmo de Lee & Lee

La idea es que dadas las soluciones obtenidas por los algoritmos para los problemas Bi-Nivel y el propuesto por Lee & Lee (2010), se parte de la solución obtenida por el algoritmo de resolución para el problema Bi-Nivel para cerrar los centros con menos Costo-Efectividad, y abrir los centros que estén abiertos en la solución obtenida por el algoritmo Lee & Lee pero no en la solución obtenida por el algoritmo Bi-Nivel, asignando los nodos de personas afectadas del centro de ayuda abierto en la solución Lee & Lee (2010) pero en la solución Bi-Nivel. Si no se empeoró el Costo-Efectividad en la solución Bi-Nivel luego de abrir el centro de ayuda y asignar

los nodos de personas afectadas, entonces se prosigue con el siguiente centro de ayuda, en caso contrario, se cierra el centro abierto y se pasa al siguiente centro de ayuda abierto en la solución de Lee & Lee (2010). Esto puede mejorar la solución dado que Lee & Lee tiene mejor cubrimiento pero a mayor costo, se intentará hallar una solución intermedia.

Se ordenará por Costo-Efectividad de la siguiente forma: $\frac{Q_i}{C_i}$ donde Q_i es el cubrimiento obtenido por el centro en cuestión, y C_i es el costo de abrir dicho centro y el costo de cubrir a las personas que tenga cubiertas. Al valor Costo-Efectividad del centro i se le llamará CE_i

Entradas del algoritmo:

$SoluciónBiNivel$: La solución obtenida luego de ejecutar el algoritmo Bi-Nivel

$SoluciónLeeAndLee$: La solución obtenida luego de ejecutar el algoritmo Lee & Lee

$Problema$: Instancia del problema que se utilizó para resolver el algoritmo Bi-Nivel, el mismo contiene todas las variables definidas en el modelo matemático

Variables:

LBI: Lista de centros abiertos en un cierto nivel (Bi-Nivel)

LLL: Lista de centros abiertos en un cierto nivel (Lee & Lee)

J^*: Nodos de personas afectadas a asignar

$Nivel$: Valor del nivel actual

Funciones:

$CentrosAbiertosEnNivel(Solucion, Nivel, Orden)$: Dado una solución y un nivel, retorna todos los centros de ayuda que estén abiertos para el nivel pasado como parámetro, ordenado por costo efectividad en el orden indicado (descendente o ascendente).

$Cerrar(Centro, Solución)$: dado una solución y un centro de ayuda, cierra el centro de ayuda en la solución pasada como parámetro, guardándolo en la Solución.

$Abrir(Solución, Centro)$: dada una solución y un centro de ayuda, abre el centro de ayuda en la solución pasada como parámetro, guardándolo en la Solución.

$AsignarClientes(Solución, Cliente, Centro)$: Dada una solución, un centro de ayuda y un nodo de personas afectadas, asigna el nodo de personas afectadas al centro de ayuda pasado como parámetro, guardándolo en la solución pasada como parámetro.

$Asignaciones(Solución, Centro)$: Dada una solución y un centro de ayuda, retorna todos los nodos de personas afectadas asignados al centro de ayuda correspondiente a la solución pasada como parámetro.

$CEAsignarClienteACentro(Solución, Centro, Clientes)$: Dado una solución, un centro de ayuda y una lista de nodos de personas afectadas, retorna el costo efectividad de asignar los nodos de personas afectadas al centro de ayuda pasado como parámetro.

$DesasignarCentroCliente(Solución, Centro, Clientes)$: Dada una solución, un centro de ayuda y una lista de nodos de personas afectadas, quita la asignación de los nodos de personas afectadas que están relacionados con el centro de ayuda pasado como parámetro, guardándolo en la Solución.

Pseudocódigo:

Inicio

1. Optimización2(SoluciónBi-Nivel,SoluciónLeeAndLee,Problema)
2. $MejorSolución = SoluciónBiNivel$
3. $Nivel = 0$
4. **Mientras** $Nivel < Problema.L$ **hacer**
5. $LBI = CentrosAbiertosEnNivel(SoluciónBiNivel, Nivel, "descendente")$
6. $LLL = CentrosAbiertosEnNivel(SoluciónLeeAndLee, Nivel, "ascendente")$
7. **Para cada** $i \in LBI$ **hacer**
8. $Cerrar(i, SoluciónBiNivel)$
9. **Para cada** $il \in LLL \mid il \notin LBI$ **hacer**
10. $Abrir(SoluciónBiNivel, il)$
11. $J^* = Asignaciones(SoluciónLeeAndLee, il)$
12. $AsignarClientes(SoluciónBiNivel, J^*, il)$
13. **Si** $CEAsignarClienteACentro(MejorSolución, il, J^*) > CEAsignarClienteACentro(SoluciónBiNivel, il, J^*)$ **entonces**
14. $MejorSolución = SoluciónBiNivel$
15. **Si no**

16. $DesasignarCentroCliente(SoluciónBiNivel, il, J^*)$

17. **Fin Si**

18. **Fin Para**

19. **Si** $MejorSolución <= SoluciónBiNivel$ **entonces**

20. $Abrir(SoluciónBiNivel, i)$

21. **Fin Si**

22. **Fin Para**

23. $Nivel = Nivel + 1$

24. **Fin Mientras**

25. **Retorno** $MejorSolución$

26. FIN.

En las líneas (5) y (6) se obtienen los centros de ayuda de manera descendente, de esa forma, al momento de iterar en las líneas (7) y (9), el primer elemento de la iteración será el centro de ayuda con menor costo efectividad de la lista LBI y LLL.

En la línea (19) comprueba si en la iteración anterior no se encontró una mejor solución a la solución actual, entonces se vuelve a abrir el centro de ayuda que fue cerrado en la línea (8).

3.2.4. Combinación de algoritmos

A modo de obtener una mayor cantidad de métodos de resolución al problema descrito, se decidió combinar ambas optimizaciones y comparar los resultados obtenidos. Por lo tanto las mismas instancias que se resolverán mediante el algoritmo de resolución para el problema Bi-Nivel, también se resolverán mediante la optimización uno, luego la optimización dos, y por ultimo una combinación de las dos optimizaciones. En la Sección 7 de Resultados Obtenidos se entrará en detalle en este punto.

3.2.5. Problema con costos acotados

Como se explicó en las Secciones anteriores, las soluciones obtenidas mediante la resolución por el modelo Bi-Nivel serán iguales o de peor calidad que las obtenidas mediante la resolución del modelo propuesto por Lee & Lee (2010) en cuanto a cobertura obtenida. En el solver utilizado

para resolver modelos matemáticos de forma exacta (GLPK) no es posible obtener la solución óptima del modelo Bi-Nivel debido a la naturaleza del mismo (ya que tiene como restricción un problema de optimización), por ende no es posible conocer qué tan cerca de la solución óptima se está al obtener soluciones mediante la ejecución de los algoritmos propuestos. Sin embargo, se sabe que la cota superior del modelo Bi-Nivel será siempre la cobertura obtenida en la solución del modelo Lee & Lee. Dicha cota puede ser que difiera mucho de la solución obtenida en el algoritmo de resolución para el problema Bi-Nivel y de la solución óptima del modelo Bi-Nivel (si es que existe), por lo tanto se formuló un modelo matemático que busca acortar la distancia entre ambas soluciones y permita una comparación más realista.

El modelo utilizado para la comparación es una combinación entre el modelo Bi-Nivel y el modelo propuesto por Lee & Lee (2010). La idea consiste en construir un modelo en donde su principal restricción consiste en que el costo de la solución no puede superar un valor preestablecido. Dicho valor será el costo obtenido de la solución con la cual se quiere realizar la comparación.

El modelo tiene como base el modelo propuesto por Lee & Lee (2010), agregando las restricciones de Bi-Nivel, y una restricción extra que evitará que la solución obtenida supere el costo de la solución dada por el Bi-Nivel.

La comparación consta de dos partes, la ejecución del algoritmo para obtener la solución que se desea comparar y la posterior ejecución del modelo matemático (utilizando el solver GLPK) pasándole como parámetro (además de la instancia) el costo obtenido por la solución obtenida. A continuación se describe el modelo matemático.

Parámetros:

- CTM: Costo máximo, es el costo máximo que puede tener la solución.

Modelo

$$\max_{x,y} \sum_{i\in I} \sum_{j\in J} \sum_{k\in K} c_{ijk} \times f_{jk} \times x_{ijk} \tag{24}$$

Sujeto a:

$$x_{ijk} \leq \sum_{l=k}^{L} y_{il}\,,\ \forall i,j,k \tag{25}$$

$$\sum_{i\in I} x_{ijk} \leq 1,\ \forall j,k \tag{26}$$

$$\sum_{i\in I} y_{il} \leq p_l,\ \forall l \tag{27}$$

$$\sum_{j \in J} f_{jk} \times x_{ijk} \leq t_{ik},\ \forall i,k \quad (28)$$

$$\sum_{l \in L} y_{il} \leq 1,\ \forall i \quad (29)$$

$$t_{ik} \leq M\left(\sum_{l=k}^{L} y_{il}\right),\ \forall i,k \quad (30)$$

$$\sum_{i \in I} t_{ik} \geq \sum_{j \in J} \alpha_k f_{jk}\ , \forall k \quad (31)$$

$$\sum_{k \in K} v_k t_{ik} \leq \min(E_i, \sum_{j \in J} \sum_{k \in K \wedge c_{ijk} > 0} f_{jk}), \forall i \quad (32)$$

$$\sum_{i \in I} \sum_{l \in L} S_{il} y_{il} + \sum_{i \in I} \sum_{k \in K} m_{ik} t_{ik} \leq CTM \quad (33)$$

$$x_{ijk} = \{0,1\} \forall i,j,k \quad (34)$$

$$y_{il} = \{0,1\} \forall i,l \quad (35)$$

$$t_{ik} \geq 0\ , \forall k \quad (36)$$

Las restricciones desde (25) hasta (32) y desde la (34) a (36) inclusive fueron explicadas en la Sección 3.2 de Formulación y algoritmo para problema Bi-Nivel.

La restricción (33) establece que los costos totales de la solución, no pueden superar el costo máximo (el cual es un parámetro y representa el costo total de otra solución).

3.3. Especificación del Sistema

El sistema se divide en dos partes: la resolución de problemas (implementación de los algoritmos propuestos) y el manejo de instancias y soluciones. La primera parte fue implementada en Java y su objetivo principal es la resolución de los algoritmos, para eso recibe como entrada una instancia y se retorna una solución dependiendo de qué algoritmo fue seleccionado para su resolución (para más información de su implementación ver Sección 6 de Implementación).

La segunda parte del sistema fue implementada en C# de .Net. Este sistema es el que brinda acceso a la creación, visualización, información estadística y resolución de las instancias, para eso el sistema brinda una página web para acceder a todas las funcionalidades mencionadas anteriormente. En el caso específico de resolución de instancias, este sitio web se comunica vía *Web Services* con el sitio de resolución de problemas, de esa forma el sitio web le envía la instancia del problema a resolver y en qué algoritmo, y el sitio de resolución de problemas se encarga de hallar una solución utilizando el algoritmo seleccionado, la cual es devuelta al sitio web para su visualización.

El sitio web posee un módulo encargado de comunicarse con el software GLPK (software de resolución empleado para hallar la solución de un modelo matemático de forma exacta) el cual se utiliza para obtener las soluciones de los modelos matemáticos, de esa forma también visualizar las soluciones de los modelos y poder comparar con las soluciones obtenidas de los algoritmos propuestos. Además, el sitio web cuenta utiliza una base de datos SQL Server en la cual se almacenan todas las instancias generadas y sus soluciones.

3.3.1. Administración de instancias

Una instancia consiste en información necesaria para que los algoritmos y modelos matemáticos propuestos puedan ser ejecutados pasándole como entrada dicha instancia. Por lo tanto, una instancia consta con todos los parámetros y variables definidos en el modelo matemático para problema Bi-Nivel. Estas instancias pueden ser utilizadas en todos los algoritmos ya que las variables y parámetros del modelo Bi-Nivel contienen todas las variables y parámetros de los otros modelos y algoritmos.

Es posible crear, modificar, borrar y examinar instancias. Existen dos formas de crear las instancias: manual o generación aleatoria. La primera de ellas consiste en ubicar los nodos de personas afectadas y las posibles ubicaciones de los centros de ayuda en un mapa geográfico (cuadrícula bidimensional con ejes x e y). El mapa ayudará a tener una idea de la distancia que hay entre las diferentes entidades y poder tener una idea geográfica del problema a atacar, asemejándose más a la realidad y mejorando la comprensión del problema. También es posible ingresar cualquier dato que el problema requiera:

- Servicios: Cantidad de servicios que los centros de ayuda podrán brindar, el cual puede tener un nombre y radio de cobertura total y parcial. Para el caso de los radios de cobertura del servicio, el mismo se visualizará en el mapa para tener una idea de qué centros de ayuda alcanzan a brindar dicho servicio a qué nodo de personas afectadas.
- Parámetro alfa: Se puede especificar un alfa distinto para cada tipo de suministro existente, este alfa refiere al porcentaje de la demanda que se debe preposicionar como mínimo para cada tipo de suministro. Una solución cuya cantidad de preposicionamiento entre todos los centros abiertos para un tipo de suministro no cumpla con el porcentaje definido para ese tipo se considera no factible.

- Niveles: Se pude especificar la cantidad de niveles que tendrá la instancia, como también el parámetro P_l(cantidad de centros de ayuda abiertos para el nivel l)
- Centros de ayuda: al seleccionar un centro de ayuda se podrá especificar la información de dicho centro: capacidad de almacenamiento, costo de abrir un centro en un nivel dado, costo de preposicionar una unidad de un servicio dado y un identificador para el centro de ayuda.
- Nodo de personas afectadas: al seleccionar un nodo de personas afectadas se podrá especificar la cantidad de personas que requieren un servicio dado y un identificador para el nodo de personas afectadas.

La segunda forma de crear instancias es a través del generador aleatorio, es posible definir parámetros o rangos para la generación de las instancias de forma aleatoria, por lo tanto las instancias creadas respetarán los rangos ingresados por el usuario. Se utiliza una distribución normal para la generación de los datos. Los parámetros se pueden especificar a través del sitio web o utilizando archivos de configuración especificados dentro de la solución del sitio web. Los posibles parámetros a especificar son los siguientes:

- Cantidad de centros de ayuda (indica la cantidad de centros de ayuda que tendrá la instancia generada, y se dispondrá de forma geográfica, coordenadas x e y, de forma aleatoria).
- Cantidad de nodos de personas afectadas (indica la cantidad de centros de ayuda que tendrá la instancia generada, y se dispondrá de forma geográfica, coordenadas x e y, de forma aleatoria).
- Cantidad máxima de personas por nodo de personas afectadas: indica la cantidad máxima de personas que tendrá los nodos de personas afectadas, la misma será uniformemente distribuido entre uno y el máximo ingresado. Esta distribución uniforme será por cada nodo, (dado dos nodos de personas afectadas no tienen por qué tener la misma cantidad de personas). A su vez la cantidad de personas necesitaran de un servicio el cual estarán distribuida uniformemente entre los diferentes servicios existentes, sin superar la cantidad máxima de personas ingresada.

- Cantidad de servicios: cantidad de servicios que tendrán las instancias generadas tanto para los centros de ayuda (las cuales brindan esos servicios) como para los nodos de personas afectadas que requieren ese servicio.
- Porcentaje de cobertura máximo: Porcentaje (del 1 al 100) que se está dispuesto a cubrir por los centros de ayuda para un servicio dado. Cada servicio tendrá un porcentaje de cuanto hay que cubrir por cada nodo de personas afectadas el cual el algoritmo deberá cumplir para que la solución sea factible. Cada servicio tendrá un valor distribuido uniformemente entre cero y el máximo porcentaje ingresado.
- Volumen máximo por paquete: cada unidad de servicio posee un volumen. Este parámetro indica cuanto será el máximo volumen de un servicio, el cual estará distribuido uniformemente entre uno y el volumen máximo por paquete ingresado.
- Cantidad de niveles: cantidad de niveles que tendrán las instancias generadas.
- Radio de cobertura parcial máximo: unidades métricas que un centro de ayuda puede cubrir parcialmente. Si un nodo de personas afectadas se encuentra ubicado dentro del radio de cobertura parcial máximo, entonces el valor de la cobertura al asignar ese nodo de personas afectadas a dicho centro de ayuda no será total, sino que será un cierto porcentaje el cual dependerá de la distancia que estén ente ellos. Mientras más lejos estén, más se acerca a cero. El radio es asignado por servicio, el cual será distribuido uniformemente entre el cero y Radio de cobertura parcial mínimo.
- Radio de cobertura total máximo: unidades métricas que un centro de ayuda puede cubrir. Si un nodo de personas afectadas se encuentra ubicado dentro del radio de cobertura total máximo, entonces el valor de la cobertura al asignar ese nodo de personas afectadas a dicho centro de ayuda será máximo. El radio es asignado por servicio, el cual será distribuido uniformemente entre el Radio de cobertura parcial mínimo y el radio de cobertura total máximo.
- Costo máximo de abrir un centro e ayuda en un nivel: Costo por abrir un centro de ayuda en un nivel dado. El valor es uniformemente distribuido entre el cero y el valor ingresado.
- Volumen máximo por centro de ayuda: Máxima capacidad de almacenamiento de un centro de ayuda, va desde uno hasta el valor ingresado.
- Cantidad máxima de centros de ayuda abiertos: valor máximo de centros de ayuda permitido abrir para que una solución sea factible. Se distribuye de forma uniforme por nivel, siendo de cero al máximo ingresado.

- Costo máximo de instalar una unidad de servicio en un centro de ayuda: Consto máximo que tiene cada unidad de servicio al ser preposicionados en un centro de ayuda. El valor se distribuye de forma uniforme entre uno y el máximo ingresado.

Una vez finalizada la especificación de los valores para la creación de la instancia, es posible darle un nombre y una descripción de forma de que se persista en la base de datos para una posterior reutilización.

Se cuenta con la posibilidad de modificar una instancia ya creada, seleccionándola dentro de una lista. Al seleccionar se dibujará en el mapa la instancia seleccionada y será posible modificar/borrar cualquier parámetro que existente. También es posible borrar la instancia completa.

En la página web se puede seleccionar una instancia y hallar su solución, para esto se puede seleccionar cualquier algoritmo o modelo matemático de los implementados. Una vez seleccionado el método de resolución, el sitio web deberá invocar los servicios brindados por el sistema de resolución de algoritmos (sistema Java) o al proceso GLPK en caso de querer ejecutar algún modelo matemático. Una vez obtenida la solución, esta se dibujará en el mapa, cambiando de color los centros de ayuda abiertos, y los nodos de personas afectadas que fueron asignados a un centro de ayuda. Al seleccionar un centro de ayuda abierto en la solución obtenida, se podrá visualizar en el mapa (con diferente color) todos los nodos de personas afectadas asignado a dicho centro, así como también la cantidad de servicios preposicionados en el centro seleccionado. Se dispone del mismo funcionamiento cuando se selecciona un nodo de personas afectadas, mostrando con otro color aquellos centros de ayuda que le están brindando algún tipo de suministros.

El sistema también cuenta con la posibilidad de descargar el código para ejecutar la instancia seleccionada en los solvers utilizados para solucionar un modelo matemático de forma exacta, los solvers que se pueden elegir son GLPK o AMPL, en cada caso el código generado será tal que cumpla con el lenguaje del solver. Con esta funcionalidad se da la posibilidad de hacer un estudio independiente del sistema, pero utilizando la misma instancia.

Las instancias pueden ser exportadas, y posteriormente importadas. Esta exportación generará un archivo que contiene toda la información de la instancia, la utilidad es que pueda ser importada en otros sistemas a modo de poder compartir las instancias para un análisis grupal. Con esto lo que se gana es poder intentar resolver las diferentes instancias bajo distintos hardware, pudiendo comparar así por ejemplo tiempos de resolución.

4. Diseño

El sistema consiste en tres partes principales, la visualización y generación de datos, la resolución de los algoritmos propuestos y el solver GLPK. A continuación se explica la arquitectura del sistema. Esta se divide en dos, primero la arquitectura para la visualización y generación de datos y luego la arquitectura para la resolución de los algoritmos propuestos. Cabe destacar que el solver GLPK es un sistema de terceros el cual se integra a nuestro sistema, por lo que no se describirá su arquitectura.

4.1. Arquitectura

Se decidió utilizar un estilo arquitectónico en capas. En este esquema las capas más altas consumen servicios definidos en las capas más bajas.

4.1.1. Arquitectura Presentación y generación de datos

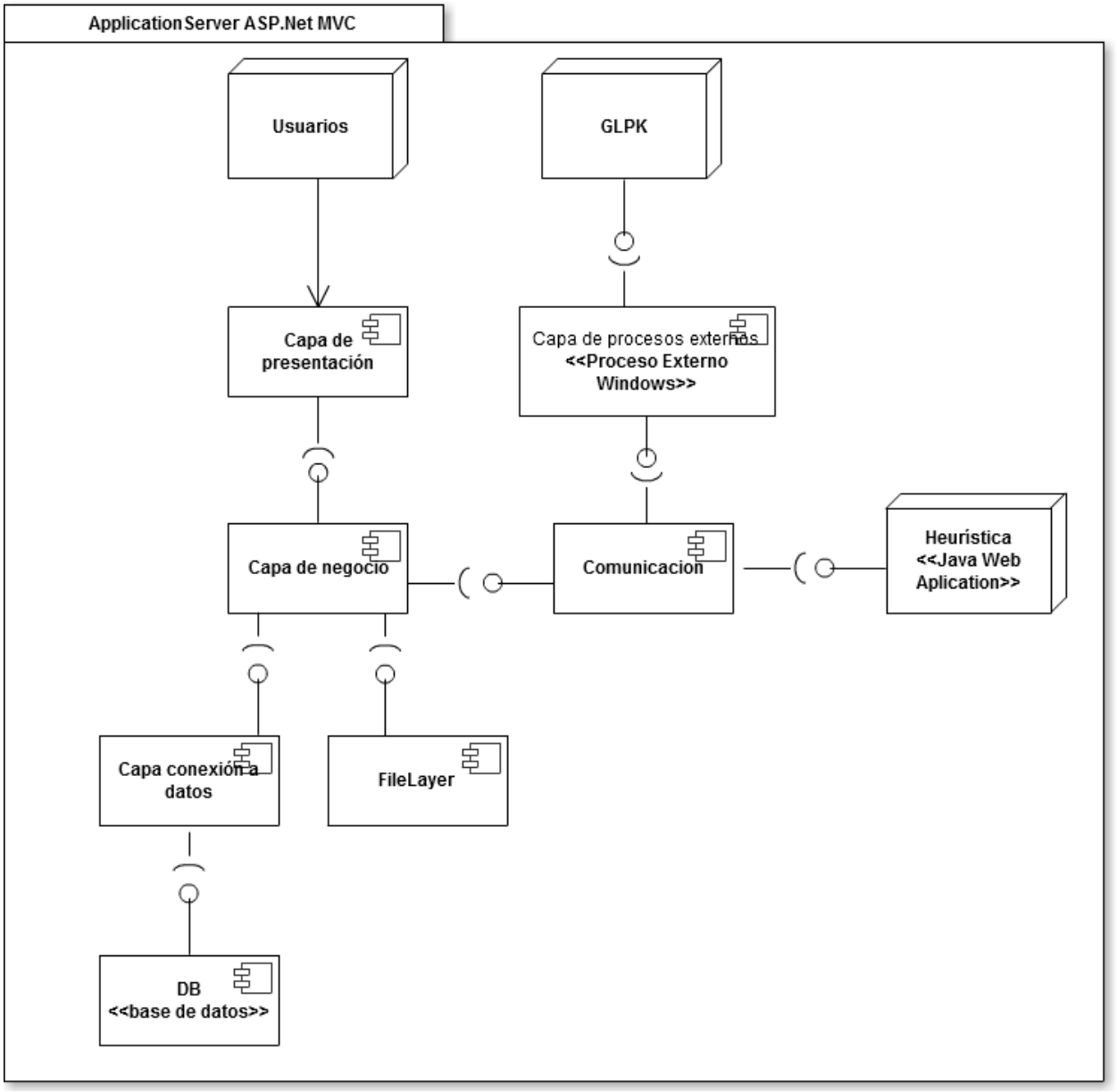

Figura 1. Arquitectura Presentación de datos

En la Figura 1 se describe la arquitectura para la presentación y generación de datos. Sus objetivos principales son generar instancias válidas para ser ejecutadas en los algoritmos y presentar de una forma amigable las instancias generadas con su solución en caso de que las tenga. También se persisten todos los datos generados y sus correspondientes soluciones para poder comparar conjuntos de instancias.

El nodo “Usuarios” representa al usuario que ingresará al sistema y podrá interactuar con el mismo a través de la componente “Capa de presentación”.

El componente “Capa de presentación” se encarga de procesar los datos ya calculados, obtenidos a través de la interfaz brindada por la componente “Capa de negocio” y presentarlos de una forma amigable al usuario. También ofrece una interfaz realizar la resolución de los algoritmos presentados en la Sección 3 de Algoritmos a desarrollar.

El componente “Capa de negocio” es el eje principal para la comunicación entre las componentes que presentan datos, la que persiste datos y la componente que obtiene las soluciones de los sistemas externos. También es la que genera instancias válidas para la resolución de los algoritmos.

Hay dos componentes encargados de persistir los datos, el primero es “Capa de conexión a datos” encargado de almacenar las instancias generadas y las soluciones de los algoritmos propuestos únicamente en una base de datos. El segundo componente es “FileLayer”, encargado de procesar archivos. Este último componente tiene como objetivo principal la transmisión de datos al sistema externo GLPK y la persistencia de las soluciones generadas por el mismo. La comunicación entre el sistema y GLPK es a través de archivos. El sitio web se encarga de convertir los datos de la instancia seleccionada a un archivo de texto que luego será interpretado por el software GLPK. Una vez creado el archivo, se le indica al sistema operativo que ejecute el proceso GLPK, pasándole como parámetro la ruta del directorio en el cual se encuentra dicho archivo y la ruta en donde GLPK debe generar el archivo para almacenar la solución. Al finalizar la ejecución del solver GLPK el sitio web se encarga de tomar el archivo generado y convertirlo en un objeto que representa la solución para su posterior visualización.

El componente “Comunicación” es el encargado de comunicarse con el sistema externo GLPK y con el sistema encargado de la resolución de los algoritmos propuestos (nodo “Heurística”)

El componente “Capa de procesos externos” es el encargado de crear procesos externos en particular para levantar el sistema GLPK.

4.1.2. Arquitectura Resolución de algoritmos.

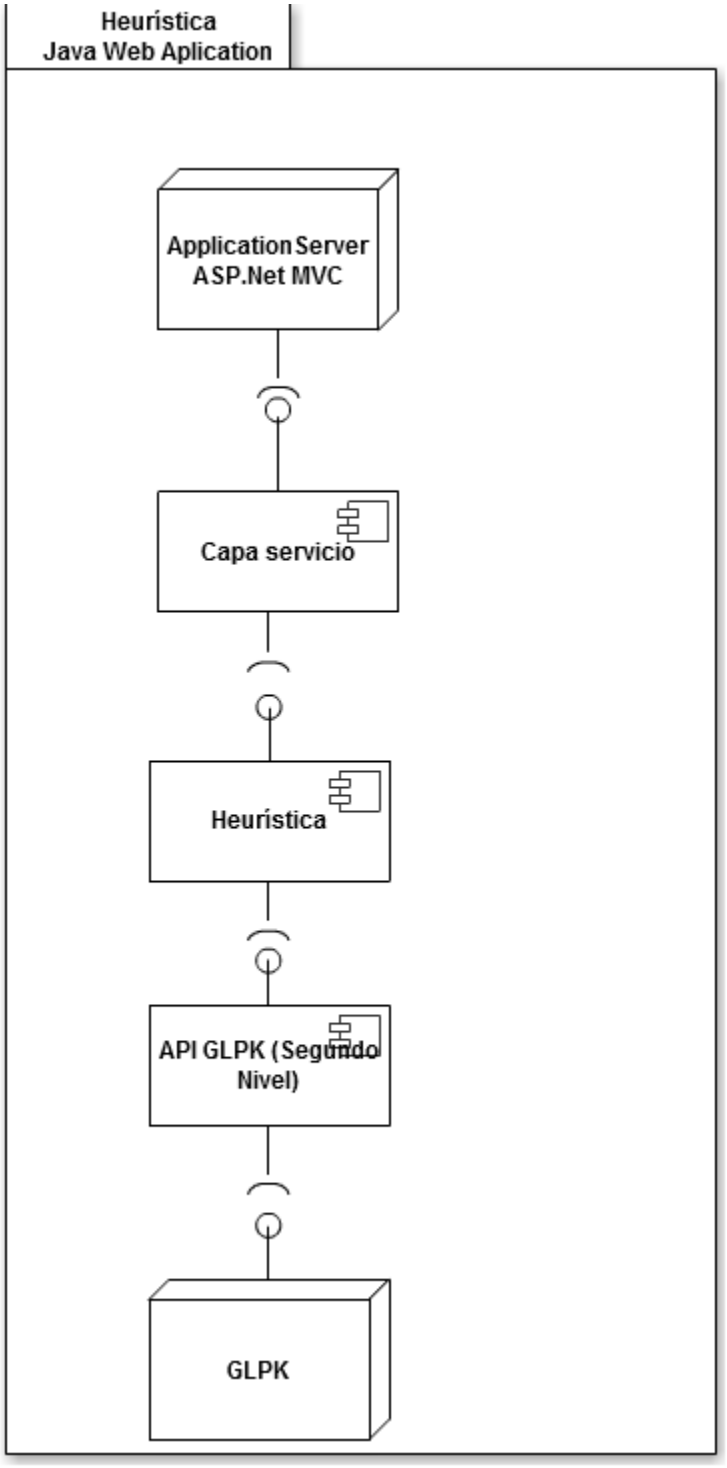

Figura 2. Arquitectura Resolución de algoritmos

En la Figura 2 se muestra la arquitectura encargada de resolver los algoritmos propuestos. Posee dos componentes principales que son “Heurística” (en donde se resuelven los algoritmos)

y “Capa de servicio” (donde brinda los servicios para la elección de los algoritmos a utilizar y los datos a consumir, es decir las instancias).

En el componente “API GLPK” se utilizan las APIs de GLPK para la resolución del segundo nivel del problema de programación Bi-Nivel. La API es la encargada de ir a los módulos de GLPK instalados en el sistema e invocar al solver pasándole como parámetro la instancia a resolver junto con el modelo matemático que se quiere utilizar.

4.2. Subsistemas: Diagrama de componentes

4.2.1. Capa de negocio

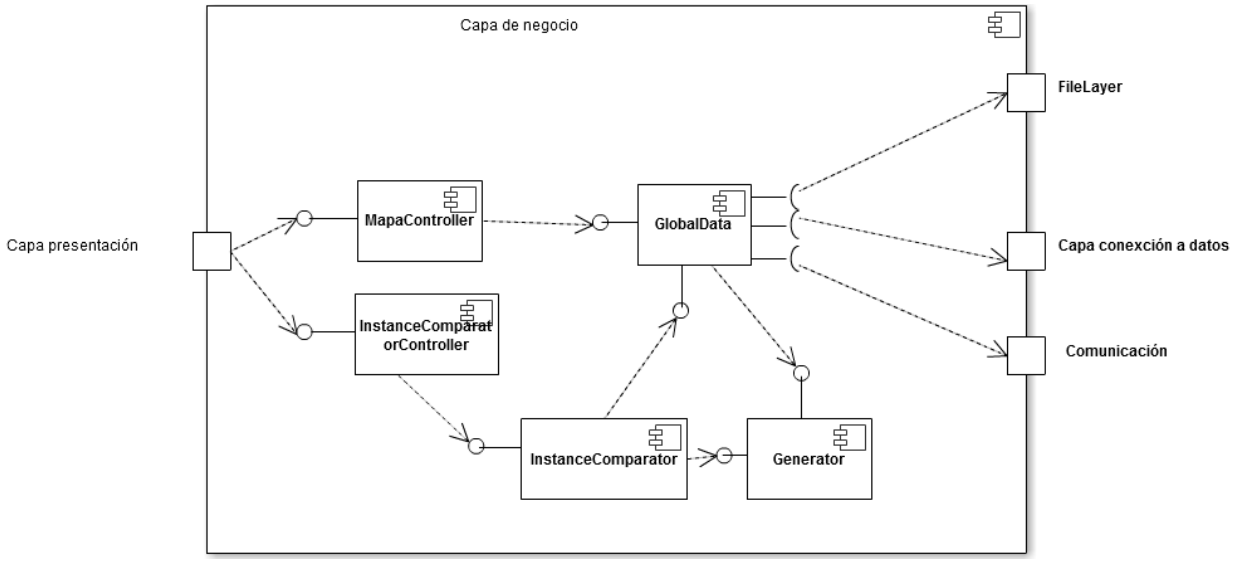

Figura 3. Diagrama de componentes capa de negocio

La capa detallada en la Figura 3 es la encargada de implementar la funcionalidad principal para la generación de instancias, encapsula la lógica de negocio relevante para la aplicación. Consiste en componentes que exponen interfaces para que otros utilicen. Este componente controla el acceso a los servicios de negocio desde otras capas, la publicación de los servicios de negocio mencionados e invocación de la capa de persistencia. Se distinguen cinco componentes principales en la capa de negocio:

MapaController: Encargado de recibir los dato de las instancias y convertirlo en coordenadas geográficas para su visualización. Dispone los datos de cada instancia para su creación y edición.

InstanceComparatorController: Brinda servicios para mostrar aquellas instancias con iguales características para comparar sus soluciones de manera eficiente y resumida. Acá es donde se

le da importancia a los resultados obtenidos, principalmente los tiempos de ejecución, la cobertura obtenida y los costos obtenidos por cada algoritmo.

GlobalData: Su función principal es mantener un lugar centralizado para los datos de las instancias manteniendo un cache para su rápido acceso. Es muy importante mantener los datos almacenados en un lugar de rápido acceso ya que los tiempos de ejecución de los algoritmos pueden ser muy grandes, como también la cantidad de datos que posee cada instancia.

InstanceComparator: Procesa todas las instancias almacenadas y las agrupa según sus características. Se considera que dos instancias poseen iguales características si ambas poseen los mismos valores en sus parámetros (cantidad de centros de ayuda, cantidad de servicios, cantidad de nodos de personas afectadas, cantidad de niveles, misma cantidad de centros de ayuda abiertos por nivel). Para una mejor definición ver Sección 4 de Especificación funcional).

Generator: Genera instancias de forma aleatoria. Es posible especificarle la estructura de la instancia a partir de parámetros del sistema o por los ingresados por el usuario (ver Sección 4 de Especificación funcional).

4.2.2. Capa conexión a datos

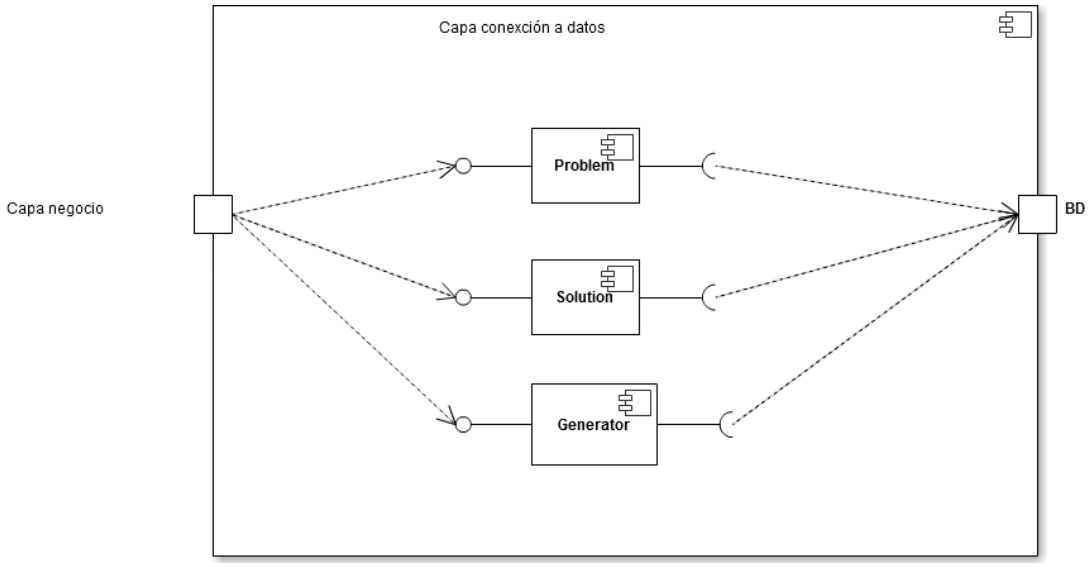

Figura 4. Diagrama de componentes capa de conexión a datos

Esta capa se encarga de gestionar los datos de la base de datos. Como se puede ver en la Figura 4, se destacan tres componentes principales:

Problem: Accede a todos los datos relacionado a las instancias ya generadas. No interactúa con las soluciones de dichas instancias.

Solution: Maneja el acceso a la base de datos para la obtención de las soluciones de las instancias. Únicamente almacena las soluciones de los algoritmos propuestos, no así las soluciones generadas por el solver.

Generator: Utilizado en el momento de generación de instancias, obtiene información relacionada con las otras instancias para poder clasificar las instancias según los criterios definidos en la Sección 3 de Algoritmos a desarrollar. Este componente tiene especial relevancia para el acceso rápido de la información de las otras instancias almacenadas, donde únicamente se quiere información general para poder clasificar otras instancias.

4.2.3. Vista de distribución

A continuación se presenta la vista de distribución del sistema propuesto.

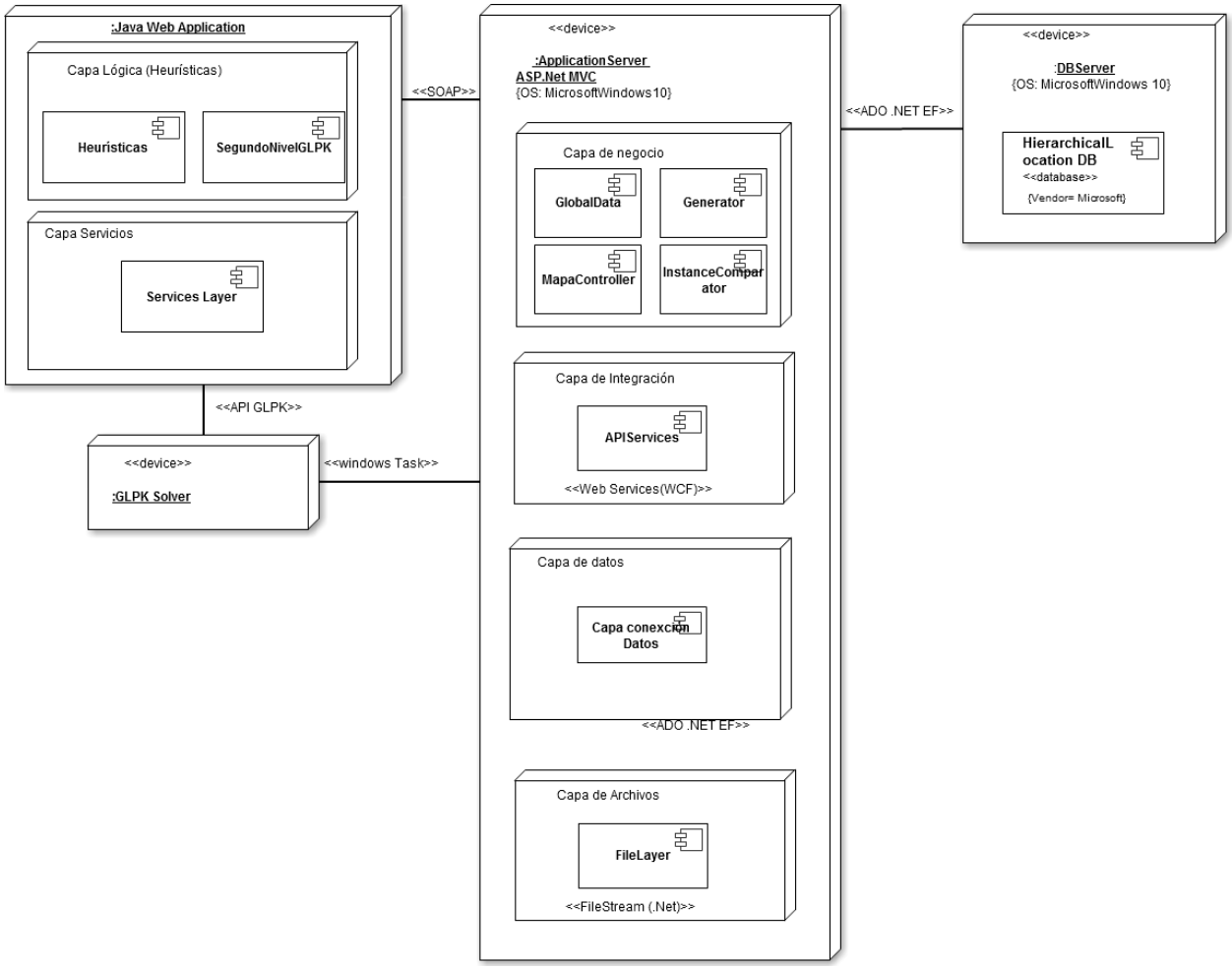

Figura 5. Vista de distribución del sistema

Como se ve en la Figura 5, el sistema consta de cuatro nodos principales:

Application Server: Lugar en donde el usuario interactúa directamente con el sistema permitiendo ejecutar las funcionalidades implementadas. También cumple la función de generador de datos.

Java Web Application: Aquí se encuentran implementados los algoritmos propuestos.

GLPK Solver: Resuelve problemas de programación matemáticos. Brinda una API para ser utilizada desde el sistema Java y a través de las tareas de Windows se ejecuta desde el sistema Application Server.

DB Server: Manejador de base de datos.

4.3. Vista de implementación

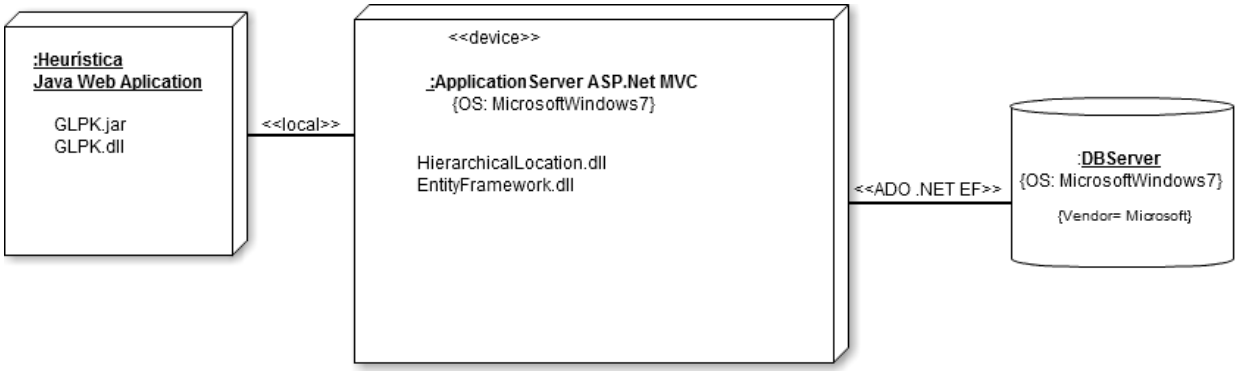

Figura 6.Vista de implementación del sistema

4.4. MER

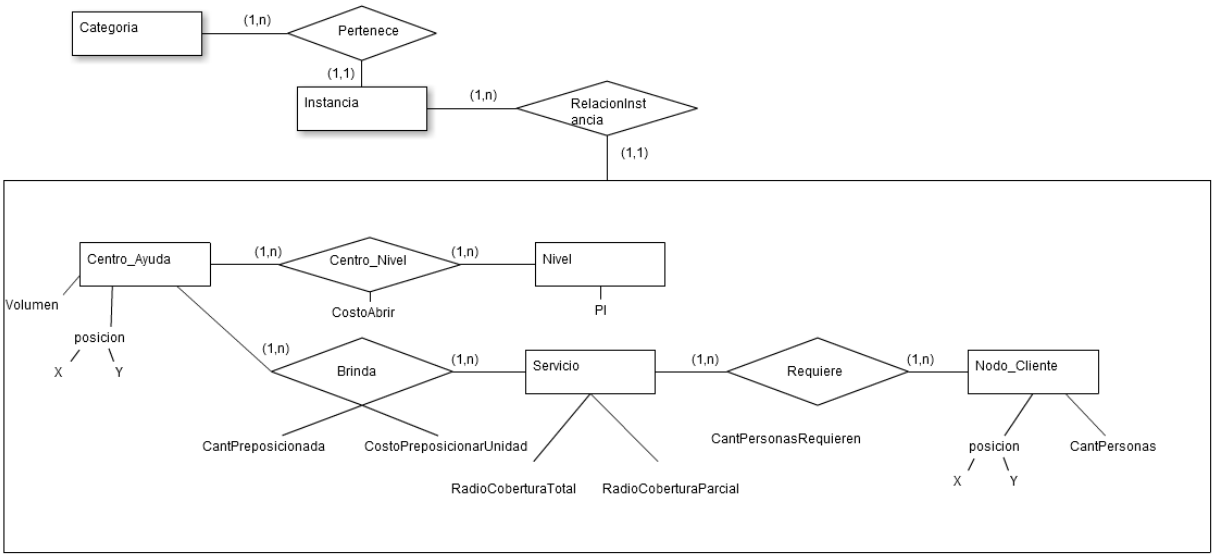

Figura 7. Modelo Entidad Relación

En la Figura 7 se muestra el Modelo Entidad Relación del sistema, cada instancia es un conjunto de datos que representa un problema a ser atacado. Cada instancia es categorizada según el número de entidades que posea para las siguientes entidades: Centro_Ayuda, Nivel, Servicio y Nodo_Cliente. Por ejemplo, toda instancia que tenga tres Centro_Ayuda, dos Nivel, dos Servicio y cinco Nodo_Cliente es una categoría “A”.

Por lo tanto una categoría puede tener múltiples instancias y cada instancia pertenece a una única categoría. Cada instancia posee un conjunto de centros de ayuda, servicios y nodos de personas afectadas. Cada centro de ayuda posee uno o más niveles en donde a mayor nivel, mayor variedad de servicios podrá brindar el centro. Cada centro de ayuda brinda uno o más servicios. A su vez cada nodo de personas afectadas requiere uno o más servicios. En caso de que la instancia tenga una solución, los servicios que requieren un cliente deben ser los mismos servicios que el centro de ayuda brinda y que el nivel en el cual el centro de ayuda está abierto esté en la relación Centro_Nivel.

5. Implementación

Los algoritmos propuestos fueron implementados y ejecutados en el sistema operativo Windows 10. Además, la solución que incluye la implementación de los algoritmos propuestos fue implementarla en Java 7 para agilizar la implementación con la ayuda de las librerías nativas de java. Otra razón por la cual se decidió utilizar Java para la implementación de los algoritmos fue la utilización de la API GLPK 4.60, ya que está mejor documentada y optimizada para este lenguaje de programación que para otros lenguajes actuales. Como servidor de aplicaciones se utilizó Tomcat 8 por un tema de facilidad a la hora de configurar el servidor y publicar las interfaces utilizando servicios web, siendo Tomcat un servidor que brinda lo mínimo necesario para invocar los algoritmos propuestos.

El manejador de base de datos utilizado es SQL Server R2 2012 ya que brinda un rápido acceso a los datos en sistemas Windows. Esta característica en particular es muy útil ya que se cuenta con cientos de instancias en donde cada instancia cuenta a su vez con cientos de entidades con sus propios datos.

Para la generación, presentación y almacenamiento de datos se utilizó el lenguaje C# 6. C# proporciona una fácil integración con el gestor de base de datos SQL Server y su ejecución es particularmente buena en sistemas Windows. Otro motivo por el cual seleccionamos este lenguaje es la experiencia que tenemos con el mismo.

La comunicación entre el sistema que implementa los algoritmos propuestos (Java) y el sistema de generación, presentación y almacenamiento (C#) es a través de servicios web SOAP 1.2. El motivo principal de la utilización de esta tecnología para comunicar sistemas es la buena compatibilidad y la fácil implementación en ambas tecnologías (C# y Java).

El sistema implementado en C# que genera, presenta y almacena las instancias de los problemas utilizados es un sitio web implementado con la tecnología MVC 5 y que utiliza IIS 7 (Internet Information Services) como servidor de aplicaciones, IIS da un excelente rendimiento en sistemas operativos Windows.

En la próxima Sección 6.1 de Interfaces se explican las interfaces para la utilización de los algoritmos.

5.1. Interfaces

Como se explicó anteriormente, el sistema de resolución de algoritmos está implementado en el lenguaje Java. Este sistema brinda diferentes interfaces a través de *Web Services* SOAP en donde es posible pasarle una instancia de un problema a resolver y el algoritmo con el cual resolverlo, y que nos retorne su solución. El sistema brinda siete servicios, cada uno representa una de las alternativas implementadas para resolver este tipo de problemas:

1. Algoritmo propuesto por Lee & Lee (2010)
2. Algoritmo Bi-Nivel
3. Primer optimización para el algoritmo Bi-Nivel
4. Segunda optimización para el algoritmo Bi-Nivel
5. Optimización 2 + optimización 1 para el algoritmo Bi-Nivel
6. Modelo relajado para el problema Bi-Nivel
7. Modelo GLPK Problema de costos acotados

Cada método recibe como parámetro una instancia del problema que se quiere resolver. Dicha instancia es un objeto que tiene como atributos cada uno de los parámetros definidos en el modelo matemático Bi-Nivel. El mismo objeto se utiliza como parámetro de entrada para los diferentes métodos de resolución, y se tomarán en cuenta solamente aquellos atributos que sean de utilidad para el problema en cuestión.

A su vez cada método retorna un objeto que representa la solución obtenida. Dicho objeto posee como atributo las variables del problema (X_{ijk}, Y_{il}, T_{ik}, ver Sección 4 de Especificación Funcional), el tiempo en milisegundos de resolución del algoritmo, el resultado de la función objetivo (la cobertura obtenida) y el costo de la solución. Las variables de la solución solamente estarán cargadas en caso de corresponder en el contexto del problema atacado, por ejemplo, para el algoritmo propuesto por Lee & Lee (2010) la variable T_{ik} estará vacía.

6. Testing

El testing se realizó en dos partes, pruebas del sistema y pruebas de los algoritmos implementados. Las pruebas del sistema consisten en probar todas las funcionalidades del sistema a nivel de usuario final, mientras que las pruebas de los algoritmos se centran en probar la validez y robustez del software resultante.

A continuación, se detallarán las pruebas realizadas para el sistema y para los algoritmos propuestos.

6.1. Pruebas del sistema

Estas pruebas se dividen en dos categorías, pruebas de integración y pruebas funcionales.

Las pruebas de integración consisten básicamente en la coexistencia entre múltiples sistemas. En la solución propuesta, se tienen dos sistemas implementados con diferentes tecnologías. Uno de ellos (y el más importante) es el servicio en donde se ejecutan todos los algoritmos mencionadas en este trabajo, el mismo brinda diferentes servicios para acceder a los algoritmos. El segundo sistema (implementado en .Net) ofrece una interfaz amigable para la creación de instancias y visualización de las soluciones de las instancias, así como también información estadística de la misma. Para eso el sistema se comunica a través de los servicios brindados por el primer sistema para poder ejecutar los algoritmos propuestos (ver Sección 6 de Implementación).

Las pruebas de integración consisten en validar que los datos viajen de forma correcta de un sistema a otro. Para eso se construyen instancias pequeñas y completas, es decir una instancia con poca cantidad de nodos de personas afectadas y centros de ayuda, pero que cada nodo y centros tenga todos sus parámetros completos y que se pueda diferenciar de un nodo de otro, de esta forma es posible comprobar que la información enviada de un sistema a otro esté completa y en orden.

Las pruebas funcionales consisten en la creación y visualización de instancias e información estadística. Para la de generación de instancias se valida que los valores generados aleatoriamente sean correctos y que estuviesen dentro de un rango preestablecido. Este rango se puede definir a nivel de usuario al momento de generar las instancias (indicando el valor máximo para cada atributo, por ejemplo la cantidad máxima de personas por nodo de personas afectadas), o se pueden utilizar valores definidos en un archivo de configuración que posee el sistema. Para la validación se generaron aleatoriamente 100 instancias diferentes (50 de ellas

utilizando la configuración del sistema y las otras 50 definiéndola en la interfaz de usuario) y se comprobó una a una que los atributos de cada instancia correspondieran con los ingresados.

Como los nodos de personas afectadas y los centros de ayuda poseen una ubicación geográfica, fue posible crear un mapa en el cual se pudiera visualizar en qué posición se encuentran dichos nodos y centros, para esto se utilizó canvas de HTML5 aprovechando la tecnología web. Utilizando un mapa se tiene un mejor concepto de cómo es la instancia y que centros de ayuda logran cubrir a qué nodos de personas afectadas. Además, al seleccionar cualquier elemento del mapa es posible visualizar información de la misma, como por ejemplo la cantidad de personas en un nodo de personas afectadas, servicios requeridos por el nodo, entre otros. Las pruebas consistieron en generar instancias y comprobar que los datos que se persistían en base de datos fueran los visualizados en la página web.

El sitio web brinda una vista en la cual se puede ver información resumida de múltiples instancias, agrupadas por su categoría. Dicha información fue validada a través de ejemplos hechos a mano ya que su funcionamiento no es el foco del trabajo realizado.

6.2. Pruebas de algoritmos propuestos

Las pruebas realizadas fueron las mismas para todos los algoritmos. Una vez implementado el algoritmo se proseguía a utilizar un depurador invocando al algoritmo controlando así paso a paso qué secciones de código ejecutaba y qué resultados intermedios se iban generando. De esta forma es posible confirmar que los algoritmos hicieran exactamente lo que el pseudocódigo indicaba.

Una vez finalizada la comprobación anterior se continúa con la comparación de los algoritmos. Dicha comparación se realizó para ver que los algoritmos dieran resultados con sentido, por ejemplo, ningún algoritmo debería dar una cobertura superior que el algoritmo de Lee & Lee (2010) debido a que dicho algoritmo es una cota superior de cualquiera de los otros. Otro de los controles realizados fue que las soluciones dadas por el algoritmo Bi-Nivel no fueran superiores a los costos dados por el algoritmo propuesto por Lee & Lee (2010). En la Sección 7 de Resultados obtenidos se encuentran cómo se generaron los datos de prueba y cómo se compararon entre ellos.

7. Resultados obtenidos

En esta Sección se detallará el resultado de las pruebas realizadas. Las pruebas se dividieron en quince escenarios, en donde cada escenario cuenta con diez instancias diferentes.

Para todas las instancias se halla una solución por algoritmo. Los algoritmos utilizados son: Lee & Lee (algoritmo), Bi-Nivel, Optimización 1, Optimización 2, Combinación de algoritmos y los modelos matemáticos de Lee & Lee y GLPK Problema de costos acotados (ver Sección 4 de Especificación funcional para más información de los algoritmos/modelos matemáticos). Como resultado se tienen 15 escenarios, 150 instancias y 1050 soluciones.

La combinación de algoritmos consiste en la ejecución secuencial del algoritmo propuesto por Lee & Lee, luego el algoritmo Bi-Nivel, luego la Optimización 2 y por último la Optimización 1. La solución del algoritmo Bi-Nivel y la solución del algoritmo Lee & Lee son la entrada al algoritmo de Optimización 2, y la entrada de la Optimización 1 es la salida de la Optimización 2. Por lo tanto la solución de la combinación de algoritmos resulta ser la solución del último algoritmo ejecutado, el cual es la Optimización 1. En las tablas presentadas en esta Sección, los tiempos de ejecución de la combinación de algoritmos es la suma de los tiempos entre todos los algoritmos participantes.

Cabe destacar que en la combinación de los algoritmos, solo se utilizó la siguiente combinación: Algoritmo Lee & Lee + Bi-Nivel + Optimización 2 + Optimización 1. La combinación Algoritmo Lee & Lee + Bi-Nivel + Optimización 1 + Optimización 2 no fue realizada ya que los resultados obtenidos para esta última combinación eran de igual peor calidad que la solución obtenida en el algoritmo Bi-Nivel. Esto se puede comprobar al realizar una ejecución de la combinación para cada escenario, en la cual en todos los casos la solución obtenida de la combinación daba resultados de peor calidad. Dadas dos soluciones de la misma instancia del escenario, se define que la solución de peor calidad es aquella que tiene un valor superior de costo efectividad y a su vez tiene una cobertura inferior o un costo superior.

7.1. Generación de instancias y resultados

Todos los escenarios compartieron los siguientes parámetros:

Cantidad máxima de personas por nodo de personas afectadas = 10. Para la asignación de personas a nodos de personas afectadas se utiliza una distribución uniforme, en donde el mínimo posible es de 1.

Porcentaje de cobertura máximo = 100%. En todos los servicios se trabajó con un porcentaje del 100%. Esto quiere decir que la intención es cubrir el 100% de la demanda.

Volumen máximo por unidad de servicio = 10.

Volumen máximo por centro de ayuda = 3000. Para la asignación de los volúmenes (tanto para una unidad de un tipo de suministro como para la capacidad de un centro de ayuda) se utiliza una distribución uniforme en donde el mínimo volumen para un servicio es de 1, y el mínimo volumen para un centro de ayuda es de 400.

Para el costo de los centros de ayuda se utiliza una distribución uniforme en donde el mínimo costo es de 500. Esta distribución tiene la particularidad que a mayor nivel, mayor será el costo asociado, esto se debe a que se considera que si un centro de ayuda es abierto en el nivel n, entonces este centro podrá brindar los servicios 1, 2, 3,.., n. Por lo tanto abrir un centro de nivel i debe ser más costoso que abrir un centro de nivel $i - 1$.

Costo máximo de preposicionar una unidad de servicio en un centro de ayuda = 10, se utiliza una distribución uniforme en donde el mínimo costo es de 1.

Dentro de los resultados obtenidos, los valores a los que se le dio mayor prioridad y por lo tanto fueron el eje central para el estudio de los algoritmos fueron:

- Cubrimiento obtenido (Q): Una de las variables más importantes. Representa un número que nos indica la asignación de suministros a los nodos de personas afectadas, a mayor cobertura mayor es la demanda cubierta de los nodos de personas afectadas.
- Costo (C): Costo monetario de abrir todos los centros que indique la solución obtenida como también el costo de preposicionar una cierta cantidad de servicios. Los algoritmos propuestos se centran en hallar soluciones al menor costo posible siempre y cuando cumplan con el preposicionamiento esperado (definido con el parámetro α).
- Costo/Efectividad (CE): Mide que tan efectiva es la solución encontrada. Es el cociente entre el costo y la cobertura obtenida. Por lo tanto, si se toma esta medida para comparar soluciones, la mejor solución entre dos instancias será aquella que tenga menor valor de Costo/Efectividad.

- Tiempo: Es el tiempo que se tarda en encontrar una solución. Al estar resolviendo problemas NP-Hard el tiempo es una variable decisiva ya que no se puede asegurar que un software de resolución pueda resolver una instancia del problema de forma exacta en un tiempo razonable, por lo tanto uno de los objetivos principales de los algoritmos propuestos es encontrar soluciones que no superen los quince minutos de procesamiento.

La comparación entre las instancias fue únicamente entre instancias del mismo escenario y analizando aquellas variables con mayor prioridad (cobertura, costo, CE y tiempo) tomando en cuenta el contexto del escenario. Por ejemplo, para las instancias muy grandes la variable tiempo crece en prioridad, en cambio para instancias chicas lo más importante serían la cobertura y el costo efectividad obtenidos.

Los modelos y algoritmos descriptos fueron ejecutados en un procesador Intel Core i5 de 2.6 GHz y 6 GB de memoria RAM. A continuación se presentan las tablas con un resumen de los resultados obtenidos.

En la columna “Problema” se describe el tipo de la instancia, por ejemplo 15c20p2n2-4 refiere al escenario cuyas instancias poseen 15 posibles ubicaciones para los centros de ayuda, 20 nodos de personas afectadas y 2 niveles, en donde el primer nivel está permitido abrir como máximo dos centros de ayuda y en el nivel dos se permite abrir como máximo 4 centros de ayuda (2-4).

Las columnas Gap representan el porcentaje de la diferencia entre dos valores, por ejemplo, el Gap Q es el porcentaje de la diferencia que hay entre el cubrimiento de GLPK y del algoritmo correspondiente al Gap Q. Cuando el Gap es negativo, significa que ese valor del Gap está a un cierto porcentaje de diferencia con respecto al Gap del GLPK.

Para las tablas 1 y 2, todas las columnas Gap X es la columna X del GLPK, por ejemplo Gap CE compara el Costo-Efectividad del algoritmo en cuestión contra el Costo-Efectividad obtenido por GLPK en esa instancia. Para la tabla 3, el Gap es contra las columnas del modelo GLPK Problema de costos acotados.

	GLPK Lee & Lee				Alg. Lee & Lee				Alg. Bi Nivel				Alg. Bi Nivel + Opt 1				Alg. Bi Nivel + Lee & Lee + Opt 2				Alg. Bi Nivel + Lee & Lee + Opt 2 + Opt 1				GLPK problema de costos acotados			
Problema	Cubrim.	Costo	CE	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)
15c20p1n6	57.3588	4650.83	84.2291	344.1	-3.3693	3.75499	0.49927	838.6	-14.354	21.0655	8.38229	1250.5	-9.915	19.2488	10.9302	1270.4	-12.436	19.8639	8.82935	2116.2	-7.8058	16.8854	10.1963	2119.3	-4.2778	19.8058	16.9396	216
15c20p2n2-4	54.9619	4689.26	92.1617	152.4	-2.3891	-1.0702	-3.627	1403	-23.15	22.3927	-1.5252	1992	-15.29	16.525	1.15608	2008.3	-16.398	20.9515	4.26462	3443	-13.145	19.1039	6.34522	3548.4	-5.1834	27.4375	23.9407	399.7
15c20p3n1-2-3	55.0394	11882.3	219.813	217.4	-4.8337	4.21257	-0.9335	3713.2	-24.773	25.8768	0.1225	6300.5	-14.459	21.9261	8.46314	6320.2	-19.921	23.5833	2.75708	10060.1	-11.403	21.8659	11.0167	10069.1	-3.6204	28.4545	25.3861	616.5
20c50p1n6	158.49	7797.44	49.6973	220.5	-2.2435	4.75706	2.53549	1636.8	-17.289	34.0186	20.2602	2300.3	-12.789	30.1504	20.3428	2331.9	-14.906	33.4304	21.8337	4036	-10.975	28.9372	21.1076	4049.9	-8.4437	34.8953	29.4793	3087.2
20c50p2n2-4	144.147	10110.6	70.5608	978.4	-3.5614	3.2443	-0.2821	6265.2	-26.673	26.1847	-0.3388	7616.7	-16.805	17.3554	0.4005	7679.4	-24.086	25.5745	2.24937	13946.8	-18.875	21.5437	3.63003	13974.4	-6.0971	26.8037	22.5894	87022.7
20c50p3n1-2-3	142.641	12395.1	88.3239	625.8	-3.9087	-4.9848	-9.2352	17767.6	-29.105	28.7164	-0.3153	20667.8	-15.502	18.7726	3.91508	20795.5	-25.869	31.5441	7.41672	38708	-15.885	25.6495	12.1604	38760.8	-6.9433	33.2506	28.7868	184740
20c100p1n12	318.162	13823.8	43.9954	277.9	-1.9534	-0.7684	-2.7704	3919	-19.14	37.2861	23.367	4520.9	-12.646	29.7475	20.9986	4585	-14.946	35.6639	24.8789	8596	-11.174	29.9993	22.3198	8646.7	-7.8707	35.4086	34.7094	457511
20c100p2n5-7	315.179	18946.8	60.6218	564.9	-4.7829	-2.0637	-7.412	28628.1	-33.154	45.912	19.522	28884.2	-20.871	37.1306	20.8648	29041.4	-29.614	42.156	18.4505	57848.2	-20.082	33.0938	17.2487	57959.4	-10.526	44.1724	38.6101	430866
25c100p3n3-4-5	464.621	22772.1	49.4191	1307.3	-6.3309	-4.6851	-11.85	32619.7	-42.082	52.4984	17.1609	36202.5	-21.06	36.6128	19.9164	36527.8	-32.388	41.7979	13.3011	69362.9	-17.898	30.1391	14.6904	69782.7	-13.969	43.3423	34.9052	129999
25c150p1n15	482.643	18593.6	38.8004	449.9	-1.7315	2.96633	1.23409	9069.6	-22.943	52.0654	38.2518	9410.3	-17.091	46.9999	36.5021	9612.4	-18.916	51.0131	39.6332	18942.5	-16.482	47.6855	37.5794	19119.2	-11.74	52.522	46.5702	437144
25c150p2n7-8	484.604	24271.7	50.3653	1029.5	-5.1802	1.23806	-4.2058	32790.3	-27.723	36.4653	13.2523	34083.3	-11.012	21.8943	13.0051	34425.7	-24.371	36.213	16.2791	67904.3	-11.73	25.3064	16.2288	68222.5	-6.5585	33.9025	29.8566	817005
25c150p3n3-5-7	786.12	31107.6	39.6579	16006.8	-4.3979	-7.1194	-12.091	130254	-31.519	41.7564	19.1982	128414	-14.693	32.3829	22.3356	129363	-24.264	31.4752	11.2985	260623	-13.326	22.2686	10.2553	261243	-9.2545	35.5055	34.7829	373680
50c100p1n15	494.529	18472.8	37.5343	792.1	-3.9353	3.00816	-1.0603	9850.4	-27.292	51.1394	33.6368	10471.1	-21.709	45.0982	30.462	11053.2	-23.049	49.1022	34.7115	21522.6	-19.89	44.8104	31.9156	22066.6	-27.933	53.218	39.6691	474696
50c100p2n7-8	452.087	18518.7	41.3401	57304.1	-11.233	6.66095	-5.4671	35356.3	-38.578	52.6702	21.6883	38001.8	-25.339	38.5793	17.6213	39532.7	-31.782	47.1427	21.0808	76481.8	-24.165	37.2195	16.6354	77560.5	-29.375	55.1996	37.1273	745817
50c150p1n15	718.963	19571.2	27.7576	20394.9	-8.5707	1.66	-8.0619	11445	-29.039	48.3609	28.1867	11763.1	-22.188	40.926	24.771	12503.4	-25.69	42.1475	22.8986	24774.1	-21.044	35.609	19.323	25500.7	-16.004	48.619	39.2908	149497
50c150p2n7-8	696.636	22146.8	31.8771	209918	-5.3988	3.96182	-1.6138	72053.1	-36.828	58.9482	33.9627	67006.4	-19.889	44.8386	30.371	69942.6	-26.127	47.7897	28.2379	145016	-18.395	38.5972	23.4502	146958	-35.437	61.2318	39.2575	831354
				19411	-4.614	0.9233	-4.021	24851	-27.73	39.71	17.176	25555	-16.95	31.137	17.628	26062	-22.8	36.216	17.383	51461	-15.77	29.92	17.131	51849	-12.7	39.611	32.619	320228

Tabla 1. Resultados obtenidos contra GLPK Lee & Lee

	GLPK Problema de costos acotados				Alg. Lee & Lee				Alg. Bi Nivel				Alg. Bi Nivel + Opt 1				Alg. Bi Nivel + Lee & Lee + Opt 2				Alg. Bi Nivel + Lee & Lee + Opt 2 + Opt 1			
Problema	Cubrim.	Costo	CE	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)	Gap Q (%)	Gap C (%)	Gap CE (%)	Tiempo (ms)
15c20p1n6	54.5841	3702.47	70.8063	216	1.306194481	-27.617	-24.716	838.6	-10.403	1.12893	-10.414	1250.5	-5.8672	-0.9179	-7.1787	1270.4	-8.3574	-0.0818	-9.4121	2116.2	-3.6118	-3.4076	-7.4466	2119.3
15c20p2n2-4	52.382	3397.69	68.0777	399.7	3.17517956	-45.17	-40.03	1403	-18.87	-6.8806	-33.234	1992	-10.477	-15.786	-30.496	2008.3	-11.733	-9.2984	-25.372	3443	-8.3393	-12.134	-23.103	3548.4
15c20p3n1-2-3	52.8284	8502.33	164.084	616.5	-0.720654709	-39.903	-40.229	3713.2	-21.637	-3.3515	-32.917	6300.5	-10.792	-8.6632	-21.953	6320.2	-16.592	-8.6662	-31.323	10060.1	-7.7276	-10.678	-20.629	10069.1
20c50p1n6	145.22	5108.91	35.3189	3087.2	7.311272417	-55.246	-43.742	1636.8	-9.4751	-2.5868	-13.587	2300.3	-4.7065	-8.0313	-13.532	2331.9	-6.8025	-2.415	-10.115	4036	-2.7191	-7.9126	-10.892	4049.9
20c50p2n2-4	135.783	7379.51	54.5029	87022.7	3.138306943	-38.197	-32.987	6265.2	-21.733	-0.5314	-29.406	7616.7	-11.196	-13.933	-28.784	7679.4	-18.811	-1.7023	-25.553	13946.8	-13.232	-7.0125	-23.675	13974.4
20c50p3n1-2-3	132.821	8194.36	62.0082	184739.7	3.383258088	-65.016	-58.885	17767.6	-23.817	-7.7929	-41.972	20667.8	-9.1343	-24.293	-36.779	20795.5	-20.272	-4.4593	-31.261	38708	-9.5964	-13.091	-24.931	38760.8
20c100p1n12	292.388	8961.49	30.6846	457511.4	7.321782637	-63.789	-51.665	3919	-11.804	4.74153	-8.7799	4520.9	-4.8361	-6.7723	-12.58	4585	-7.2412	1.63535	-6.7506	8596	-3.227	-6.3273	-10.064	8646.7
20c100p2n5-7	282.537	10658.2	37.2537	430866	6.797808376	-108.18	-91.525	28628.1	-25.406	2.85214	-31.433	28884.2	-11.503	-14.358	-29.593	29041.4	-21.422	-5.3027	-34.57	57848.2	-10.758	-22.292	-37.295	57959.4
25c100p3n3-4-5	398.759	12810.9	32.5558	129999.4	10.21344537	-102.58	-81.51	32619.7	-32.152	13.9821	-26.934	36202.5	-8.164	-12.965	-23.094	36527.8	-20.905	-8.6924	-36.804	69362.9	-4.005	-30.781	-35.564	69782.7
25c150p1n15	425.94	8857.42	21.1687	437143.8	11.63701475	-122.61	-98.316	9069.6	-12.69	-1.2111	-16.07	9410.3	-6.1368	-11.665	-18.93	9612.4	-8.1274	-3.6796	-13.034	18942.5	-5.3617	-10.284	-16.583	19119.2
25c150p2n7-8	454.255	16145.8	35.583	817005.4	1.796505858	-58.792	-54.475	32790.3	-22.602	3.97562	-24.057	34083.3	-4.8163	-19.402	-25.316	34425.7	-18.919	2.89426	-19.713	67904.3	-5.617	-13.843	-20.544	68222.5
25c150p3n3-5-7	714.718	20163.8	27.6888	373679.7	6.21616304	-102.85	-85.825	130254	-25.066	11.6594	-18.804	128414	-5.866	-6.7229	-13.366	129363	-17.015	-10.979	-34.839	260623	-4.1538	-31.938	-36.8	261243
50c100p1n15	363.601	8123.12	22.3325	474695.7	43.32256166	-150.04	-75.021	9850.4	5.33693	-15.904	-11.881	10471.1	14.4132	-32.248	-17.511	11053.2	11.958	-21.591	-10.042	21522.6	17.5221	-33.82	-15.007	22066.6
50c100p2n7-8	314.864	8307.03	26.4068	745816.8	31.24076996	-125.33	-69.763	35356.3	-9.4649	-9.5926	-22.722	38001.8	9.73814	-41.152	-29.43	39532.7	0.37669	-23.227	-24.12	76481.8	11.7013	-45.878	-31.282	77560.5
50c150p1n15	607.124	9876.22	17.2829	149497	10.72611219	-112.59	-90	11445	-15.126	-0.1103	-18.32	11763.1	-7.033	-15.525	-24.443	12503.4	-11.184	-14.551	-29.512	24774.1	-5.4697	-27.937	-36.042	25500.7
50c150p2n7-8	440.164	7850.12	18.5934	831353.8	54.71542761	-176.46	-75.024	72053.1	3.41076	-18.77	-13.994	67006.4	30.8397	-58.044	-20.137	69942.6	20.2429	-50.439	-23.767	145016	32.4638	-76.92	-32.022	146958
				320228	11.1852092	-84.44	-62.98	254.09	-16.19	-2.096	-23.06	25555	-3.68	-17.91	-22.71	26062	-10.57	-9.381	-23.28	51461	-2.293	-21.13	-23.95	51849

Tabla 2. Resultados obtenidos contra GLPK Problema de costos acotados

7.2. Análisis de los resultados obtenidos

En comparación contra las soluciones obtenidas por el modelo GLPK Lee & Lee, el algoritmo Bi-Nivel genera buenas soluciones en cuanto a costos (en promedio tienen un 31% de gap favorable para el algoritmo Bi-Nivel). De todas formas, esas soluciones están muy distantes en cuanto a cobertura (en promedio tienen un 27% de gap favorable para el modelo GLPK Lee & Lee). Como el objetivo que nos planteamos es maximizar la cobertura se intentó mediante las optimizaciones mejorar la cobertura obtenida por el algoritmo Bi-Nivel. Las optimizaciones propuestas son efectivas ya que cumplen con el objetivo buscado tratando mantener el costo efectividad de la solución.

En todos los algoritmos el gap de cobertura aumenta a medida que aumenta la cantidad de niveles, siendo favorable para GLPK Lee & Lee. Esto se puede ver claramente en las instancias chicas (de 15 a 20 centros de ayuda) en donde el gap de cobertura llega a aumentar un 60% aproximadamente por cada nivel. Para el resto de las instancias, la diferencia es de aproximadamente un 30% por cada nivel.

Aplicando la Optimización 1 el gap de cobertura generalmente se acorta, llegando en el mejor de los casos a un 9.9% y en el peor de los casos a un 25.3% (estos valores se obtienen de la tabla 1 y son el mejor y el peor valor de los promedios de gap de cobertura de los escenarios respectivamente), sin embargo este algoritmo causa que el costo aumente, aunque las soluciones siguen siendo menos costosas que las soluciones obtenidas por la resolución del modelo Lee & Lee mediante el solver GLPK. Otro punto a destacar de este algoritmo, es que apenas aumenta el tiempo de resolución obtenido por el algoritmo para el problema Bi-Nivel.

La forma en la que está implementado el algoritmo para la Optimización 1 evita que el costo efectividad empeore más de lo que lo hace el algoritmo Bi-Nivel, por lo tanto las soluciones obtenidas en la Optimización 1 son mejores o iguales en cuanto a costo efectividad que las soluciones obtenidas por el algoritmo Bi-Nivel.

En cuanto a la Optimización 2, la calidad de las soluciones obtenidas suelen no ser superiores a la calidad de las soluciones obtenidas por la Optimización 1. Aunque la cobertura de la Optimización 2 es ligeramente superior (menos del 5% de diferencia en promedio), el costo efectividad es superior en la Optimización 1 (una diferencia del 30% aproximadamente en promedio). Además, el tiempo de resolución es aproximadamente el doble al obtenido por el algoritmo para el problema Bi-Nivel, lo cual es de esperarse ya que esta Optimización no solo requiere la previa ejecución del algoritmo Bi-Nivel sino también requiere la previa ejecución del algoritmo Lee & Lee.

GLPK Problema de costos acotados corresponde al modelo que más se acerca a la solución óptima al problema Bi-Nivel (si es que existe) y se compara contra todos los algoritmos del trabajo (en la Sección 3.2.5 de Problema con costos acotados se explica mejor este enfoque). Cabe aclarar que hubo ocho instancias en las cuales no se encontraron soluciones en el tiempo máximo que se definió de quince minutos.

7.3. GLPK Problema de costos acotados

La comparación contra el modelo GLPK Problema de costos acotados consiste en identificar la solución de mejor Costo-Efectividad entre los algoritmos propuestos, para luego obtener su costo. El costo obtenido es el que se pondrá como parámetro de entrada a la hora de resolver la instancia del escenario tratado mediante un solver como GLPK, utilizando el modelo GLPK Problema de costos acotados. Luego, se comparan las soluciones obtenidas, pudiendo considerar la solución obtenida por el modelo GLPK Problema de costos acotados como una cota superior.

En comparación contra las soluciones obtenidas por la ejecución del modelo GLPK Problema de costos acotados mediante un solver, se puede ver que mientras mayor es la cantidad de niveles, mayor es el gap de cobertura entre la solución a comparar y la obtenida por el modelo GLPK Problema de costos acotados (favorable para GLPK Problema de costos acotados). Esto se ve con mayor facilidad en instancias pequeñas y medianas (de 15 a 25 centros de ayuda). En instancias muy grandes como por ejemplo las instancias con 50 centros de ayuda, se puede ver que el gap de cobertura se acorta en comparación contra el algoritmo que haya dado menor costo efectividad. Sin embargo, el costo sigue siendo inferior en las soluciones obtenidas por GLPK Problema de costos acotados, lo que lleva a minimizar el Costo-Efectividad, siendo así una solución de mejor calidad.

La combinación entre la Optimización 1 y la Optimización 2 es la que posee mejor cobertura (teniendo en promedio 2.29% de gap contra GLPK Problema de costos acotados), siguiéndole la Optimización 1, Optimización 2 y por último el Bi-Nivel.

En cuanto a costos, en promedio el algoritmo que generó menor costo resultó ser el Bi-Nivel (teniendo mejores costos que el GLPK Problema de costos acotados), seguido por la Optimización 2, Optimización 1 y por último la Optimización 1 en combinación con la Optimización 2.

8. Conclusiones y trabajo a futuro

8.1. Conclusiones

El objetivo del presente proyecto fue desarrollar y evaluar un algoritmo para maximizar el cubrimiento de la demanda de las personas afectadas por un desastre natural, minimizando a su vez los costos involucrados en la localización de suministros. Para cumplir con el objetivo se formuló un modelo matemático tomando como base el modelo propuesto por Lee & Lee (2010) que utiliza la técnica *Hierarchical Covering Location Problem*, y se le realizaron algunas modificaciones. La más notoria es el agregado de una restricción que consiste en un modelo de optimización encargado de minimizar los costos de instalación y de preposicionamiento, obteniendo así una formulación Bi-Nivel.

Las implementaciones de los algoritmos propuestos dieron buenos resultados de acuerdo a las comparaciones realizadas. Cabe destacar que tanto el tiempo de ejecución para instancias medianas como para instancias grandes fue inferior al tiempo de ejecución del software GLPK (solver de resolución exacta).

El algoritmo Bi-Nivel es el que da menor costo de preposicionamiento con muy buenos tiempos de ejecución. Sin embargo, no es el algoritmo más efectivo en cuanto a cobertura obtenida. Las optimizaciones mejoraron los resultados obtenidos ya que se busca una solución local a partir de una solución global. En las optimizaciones, se buscó mejorar el cubrimiento obtenido sin empeorar, o empeorando muy poco el costo efectividad de la solución.

Cada algoritmo propuesto se centra en un aspecto particular del problema. Si se desea buscar el mejor costo-efectividad, la combinación de la optimización 1 y la optimización 2 es la que da mejor resultado, aunque el tiempo de resolución es bastante grande. Si el tiempo también es importante, entonces se debe considerar la optimización 1, la cual da buenos resultados de costo efectividad apenas incrementando el tiempo de resolución del algoritmo para el problema Bi-Nivel.

Dado que con el solver utilizado no se puede resolver el problema Bi-Nivel, se buscó una alternativa para a partir de una solución obtenida por el algoritmo para el problema Bi-Nivel se obtenga otra solución que tenga mayor cobertura pero que mantenga los costos. Para eso se formuló un modelo de programación matemática que maximiza el cubrimiento pero que no permite que el costo supere cierto valor. A este modelo le denominamos GLPK Problema de costos acotados.

Al haber desarrollado el modelo GLPK Problema de costos acotados fue posible determinar una mejor cota para las soluciones obtenidas tanto por el algoritmo Bi-Nivel como por las optimizaciones implementadas. Esto fue gracias a que dicho modelo busca maximizar el cubrimiento, pero acotando superiormente el costo por el obtenido en la solución que se quiere comparar. De esta forma se obtienen soluciones en donde la cobertura es superior a las soluciones obtenidas por el algoritmo Bi-Nivel, pero son inferiores en cuanto a la cobertura obtenida en el algoritmo de Lee & Lee (2010). Se puede ver que dada una instancia para un escenario, la solución obtenida a partir del algoritmo Bi-Nivel será de peor o igual calidad a la solución óptima obtenida por la resolución de la instancia utilizando el modelo GLPK Problema de costos acotados utilizando como cota máxima para el costo el obtenido por la solución obtenida por el algoritmo Bi-Nivel.

8.2. Trabajo a futuro

A continuación se presentan posibles mejoras y puntos a atacar para reforzar lo investigado en el proyecto.

Si bien el modelo GLPK Problema de costos acotados sirvió de cota superior para el problema Bi-Nivel, una mejor comparación sería implementar un modelo basado en el problema relajado de la programación Bi-Nivel. Con esto se tendría una solución contra la cual comparar soluciones para toda instancia sin depender del costo de la solución que se quiere medir su calidad. El problema relajado para la programación Bi-Nivel se detalla en el Estado del Arte.

Otro punto a atacar es el sitio web de generación de datos, el cual se quisiera adaptar a situaciones más reales siendo a su vez más amigable para su uso. Un ejemplo es poder seleccionar la zona de trabajo y que mediante un mapa se puedan posicionar geográficamente tanto los nodos de personas afectadas como las posibles ubicaciones de los centros de ayuda. Esto se puede realizar integrando el sistema con alguna API de geo localización como Google Maps.

A modo de brindar más información acerca de las soluciones obtenidas, también se podría agregar más datos como porcentaje de personas cubiertas, cantidad de personas que no fueron asignadas a ningún centro de ayuda, cantidad de suministros brindados contra la demanda de los mismos para cada tipo de suministro, costo total preposicionado por tipo de suministro, entre otras.

Además, luego de obtenida una solución se puede hacer un análisis por centro de ayuda, indicando si aumentando los suministros preposicionados en dicho centro se puede llegar a cubrir

más personas afectadas en el rango de alcance del centro, o dando alertas en caso de que un centro tenga más suministros preposicionados de los que en realidad necesita.

9. Bibliografía

[1] S. Duran, M.A. Gutierrez, P. Keskinocak, 2011, Pre-Positioning of Emergency Items Worldwide for CARE International, *Informs Journal on Computing*, 41, 223 - 237.

[2] A. Leiras, I. Brito,E.Q. Peres, T.R. Bertazzo,H.T.Y. Yoshizaki, 2014, Literature review of humanitarian logistics research: trends and challenges, *Journal of Humanitarian Logistics and Supply Chain Management*, 4, 95-130.

[3] A. Cozzolino, 2007, Humanitarian Logistics and Supply Chain Management, *Springer-Verlag Berlin Heidelberg*, 4, 2.

[4] M. Jahre, L.M Jensen, T. Listou, 2009, Theory development in humanitarian logistics: A framework and three cases, *Management Research News*, 32, 1008–1023.

[5] P.A. Trunick, 2005, Logistics when it counts, *Logistics Today*, 46, 38.

[6] B.M. Beamon & S.A. Kotleba, 2006, Inventory management support systems for emergency humanitarian relief operations in South Sudan, *The International Journal of Logistics Management*, 17, 187 - 212.

[7] S. Chopra & P. Meindl, 2008, Administración de la cadena de suministros Estrategia, planeación y operación. *Pearson Education*, ISBN: 978-970-26-1192-9, 3-6.

[8] A. R. Akkihal, 2006. Inventory Pre-positioning for Humanitarian Operations, *Master's thesis,* Massachusetts Institute of Technology, Massachusetts.

[9] A. Apte, 2009, Humanitarian Logistics: A New Field of Research and Action, Foundations and Trends in Technology, *Information and Operations Management*, 3, 1-100.

[10] . Megiddo & A. Tamir, 1982, On the complexity of locating linear facilities in the plane, *Operations Research Letters*, 5, 194–197.

[11] N. Mladenovic, J. Brimberg, P. Hansen, J.A. Moreno-Pérez, 2006, The p-median problem: A survey of metaheuristic approaches, *European Journal of Operational Research*, 179, 927-939.

[12] N. Mladenovic, M Labbé, P. Hansen, 2003, Solving the p-center problem with Tabu Search and Variable Neighborhood Search, *Research Gate*, 42, 48-64.

[13] R. Church & C. Revelle, 2014, The maximal covering location problem, Papers of the Regional Science Association, *Papers of the Regional Science Association*, 32, 101-118.

[14] Expanding the Pre-Positioning Network of CARE International, *International Journal of Environmental Research and Public Health*, 9, 2863–2874.

[15] A. Cobham, 1969, The intrinsic computational difficulty of functions, *Logic The Journal of Symbolic Logic*, 34, 657-657.

[16] K. Leyton-Brown, H.H. Hoos, F. Hutter, L Xu, 2014, Understanding the Empirical Hardness of NP-Complete Problems, *Communications of the ACM*, 57, 98-107.

[17] J.M. Lee & Y.H. Lee, 2010, Tabu based heuristics for the generalized *Hierarchical Covering Location Problem*, *Computers & Industrial Engineering*, 58, 638–645.

[18] B. Colson, P. Marcotte, G. Savard, 2007, An overview of bilevel optimization, Annals of Operations Research, 153, 235-256.

[19] R. G. Jeroslow, 1985, The polynomial hierarchy and a simple model for competitive analysis, Mathematical Programming, 32, 146–164.

[20] E.G Talbi, 2013, A Taxonomy of Metaheuristics for Bi-level Optimization, *Metaheuristics for Bi-level Optimization,* 482, ISBN: 978-3-642-37838-6, 1-39.

Anexo 1. Problemas de Localización

Los problemas de localización, en su forma más general, se pueden describir de la siguiente manera: Un conjunto de clientes que están distribuidos espacialmente en un área geográfica demandan un cierto producto o servicio. La demanda de los clientes debe ser cubierta por una o varias instalaciones que pueden brindar los productos o servicios demandados. Luego, el proceso de decisión establece dónde se deben ubicar las instalaciones en el territorio deseado con el fin de cumplir con la demanda, tomando en cuenta los requerimientos de los clientes y las restricciones geográficas entre otros.

En varios problemas de localización, el objetivo es minimizar las distancias entre las instalaciones y los afectados (problema *p-center* o *p-median*). Otras veces lo que se quiere es buscar la cantidad mínima de instalaciones que cubran toda la demanda (*Set Covering Problem*).

Otra propiedad importante para las instalaciones está dada por su tipo. Esta propiedad especifica las características tales como capacidad, servicio y consideraciones sobre su estructura. En casos simples, los problemas de localización requieren que las instalaciones sean idénticas para poder proporcionar el mismo servicio o producto. Los problemas también se pueden diferenciar de acuerdo a si brindan uno o varios servicios, basándose en la capacidad del centro de ayuda. Algunos problemas de localización admiten instalaciones donde se considera que la capacidad de las instalaciones es ilimitada, mientras que otros buscan la mejor ubicación con una producción limitada. Por lo tanto, los problemas de localización también pueden clasificarse como capacitados o no capacitados (Apte, 2012) [9].

Se le denomina Facility Location Problem (FLP) al problema que busca la ubicación óptima para un conjunto de centros de distribución, con el fin de minimizar el costo del transporte y considerando factores como por ejemplo evadir ubicar los centros de distribución cerca de materiales peligrosos, entre otros (N. Megiddo & A. Tamir, 1982) [10].

Los problemas de localización pueden modelarse de distintas maneras, dependiendo del criterio de optimización: *P-median*, *P-center* y *Set Covering*.

P-median (Mladenovic et al., 2006) [11]

Su objetivo es minimizar la suma total de las distancias de los nodos clientes a sus localidades asignadas. Se define como un problema de programación entera de la siguiente manera:

Sea p la cantidad de centros de ayuda a abrir

Sea L un conjunto de *m* elementos (puntos de ubicación)

Sea U conjunto de *n* elementos (personas afectadas)

D la matriz $n \times m$ de distancias $d_{ij}\ i \in U, j \in L$

Sean las siguientes variables de decisión:

y_j es 1 si hay un centro de ayuda abierto en la posición j, 0 en otro caso

x_{ij} es 1 si las personas afectadas i están servidas por un centro de ayuda abierto en la ubicación j, 0 en otro caso

Se quiere:

$$(min) \sum_i \sum_j d_{ij} x_{ij} \quad (37)$$

Sujeto a:

$$\sum_j x_{ij} = 1\ , \forall i \quad (38)$$

$$x_{ij} \leq\ y_j, \forall i, j \quad (39)$$

$$\sum_j y_j = p, \quad (40)$$

$$x_{ij}, y_j\ \in \{0,1\} \quad (41)$$

Las restricciones (38) expresan que la demanda de cada persona afectada debe ser atendida por algún centro. Las restricciones (39) previenen que exista algún nodo de personas afectadas sea atendido por una ubicación en la cual no hay centros de ayuda abiertos. Las restricciones (40) indican que la cantidad de centros de ayuda abiertos debe ser exactamente p.

P-center (Mladenovic et al., 2003) [12]

Su objetivo es asignar nodos clientes dispersos geográficamente a localidades de forma de minimizar la máxima distancia entre todos los clientes a sus localidades asignadas. Se define formalmente de la siguiente manera:

Sea V conjunto de n localidades potenciales para las instalaciones.

Sea U puntos de m personas afectadas.

w_i: Pesos para los $m\ \ u_i \in U$

$d_{i,j}$: Distancia entre $u_i\ -\ v_j$ con $v_j \in V$

y_j: Variable booleana que indica si el centro de ayuda se encuentra ubicada en la posición v_j $(y_j = 1)$

x_{ij}: Indica si el nodo cliente u_i se encuentra asignado al centro de ayuda v_j

p: Número de centros de ayuda a abrir

z: Es la distancia máxima entre un cliente y su centro de ayuda más cercano.

Se quiere:

$(min)\ Z$

Sujeto a:

$$\sum_j x_{ij} = 1\ , \forall i \quad (42)$$

$$x_{ij} \leq\ y_j, \forall i, j \quad (43)$$

$$\sum_j y_j = p, \quad (44)$$

$$z \geq \sum_j d_{ij} x_i \quad (45)$$

$$x_{ij}, y_j\ \in \{0,1\}\ \forall i, j \quad (46)$$

Las restricciones (42) expresan que la demanda de cada cliente debe ser atendida. Las restricciones (43) previenen que algún cliente sea suministrado por un centro de ayuda que no esté abierto. La cantidad máxima de centros a abrir no puede superar el número *p* (44). La variable z (45) se define la distancia más grande entre un usuario y su centro de ayuda abierto más cercano.

Set Covering (Church, 2014) [13]

Dado un conjunto de zonas a cubrir, y localidades que pueden cubrir cierta cantidad de zonas, se quiere encontrar la menor cantidad de localidades que puedan cubrir esas zonas. Más formalmente, dado un conjunto de zonas (universo) y n conjuntos tal que la unión de esos n conjuntos forman el universo, se quiere encontrar la menor cantidad de conjuntos tal que la unión siga conteniendo a todas las zonas del universo. Sean los siguientes parámetros:

I: Conjunto de zonas a cubrir.

S: la distancia en la cual un nodo se considera desprotegido (puede ser elegido de forma distinta para cada nodo).

J: conjunto de sitios donde se ponen los depósitos.

x_j: 1 si hay un depósito ubicada en el sitio j, 0 en otro caso.

y_i: 1 si el nodo de demanda i es cubierto por un depósito dentro de una distancia S, 0 en otro caso.

d_{ij}: la distancia más corta del nodo i al nodo j.

a_i: cantidad de población a servir en el nodo i.

P: cantidad de instalaciones a ser ubicadas.

$N_i = \{ j \in J \mid d_{ij} <= S \}$.

El problema de *Set Covering* se puede formular de la siguiente manera:

$$(max) \sum_{i \in I} a_{ij} y_i \quad (48)$$

Sujeto a:

$$\sum_{j \in N_i} x_j \geq y_i \quad (49)$$

$$\sum_{j \in J} x_j = P \quad (50)$$

$$x_j \in \{0,1\} \forall j \in J \quad (51)$$

$$y_i \in \{0,1\} \forall i \in I$$

Hierarchical Covering Location Problem

El objetivo principal del *Hierarchical Covering Location Problem* típico es determinar la ubicación de las instalaciones en una red de varios niveles en un modo de servir a las personas afectadas en el nivel más bajo de la jerarquía de forma eficiente (con el objetivo de reducir el costo al mínimo) y eficaz (maximizando la cantidad de personas afectadas a servir). Sin embargo, en el problema a atacar, se considera que las necesidades de una persona o comunidad pueden ser cubiertas de forma parcial o total y también se limita la cantidad de centros que será posible abrir. Se tendrá para ello una función de cubrimiento que tendrá tres parámetros de entrada: distancia entre la persona y el centro de ayuda, un valor de distancia mínima y un valor de distancia máxima. La distancia máxima define que si la distancia entre la persona y el centro de ayuda es mayor a ese valor, entonces el cubrimiento será 0, es decir que la persona no está cubierta por ese centro de ayuda. El parámetro de distancia mínima indica que si la distancia entre esa persona y el centro de ayuda es menor o igual a ese valor, entonces el cubrimiento será 1, es decir, está cubierto totalmente. En cualquier otro caso el cubrimiento será parcial, es decir un valor entre 0 y 1.

Problema Bi-Nivel

Básicamente se le llama Optimización Bi-Nivel o problema Bi-Nivel a un programa matemático que contiene un problema de optimización en sus restricciones.

$\min_{x \in X} F(x, y)$

Sujeto a:

$G(x, y) \leq 0$

$min_{y \in Y} f(x, y)$

$sujeto\ a$:

$g(x, y) \leq 0$

$x, y \leq 0$

En donde $x \in \mathbb{R}^{n_1}$ e $y \in \mathbb{R}^{n_2}$.

El programa anterior tiene un forma de jerarquía, lo cual es una característica particular de los problemas Bi-Nivel, incluyen dos problemas matemáticos en una sola instancia (uno de esos problemas es parte de las restricciones del otro problema matemático).

Viendo esta relación de herencia, se le denomina a la parte superior problema de nivel superior (en donde se le llama función objetivo de nivel superior a $F: \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$), mientras que el hijo se le denomina "problema de nivel inferior" (a la función objetivo se le llama función objetivo de nivel inferior $f: \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$). Por ende también se dividen en dos clases a las variables y las restricciones, variables de nivel superior ($x \in \mathbb{R}^{n_1}$), variables de nivel inferior ($y \in \mathbb{R}^{n_2}$), restricciones de nivel superior $G: \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$ y restricciones de nivel inferior $g: \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$.

Otra característica particular de estos problemas es que las restricciones de nivel superior involucran variables de ambos problema y juegan un rol muy específico (B. Colson et al. 2007) [18].

En los problemas Bi-Nivel no necesariamente hay una solución. Restringiendo las funciones y restricciones $F, G, f\ y\ g$ sean continuas y acotadas no garantizan la existencia de una solución.

Otra característica de estos problemas es que el orden en que se plantea es importante para su resolución (no es simétrico) (B. Colson et al., 2007) [18].

Relajación del problema Bi-Nivel.

El problema relajado de un problema de programación Bi-nivel luce de la siguiente manera (Talbi, 2013) [20]:

$\min_{x \in X} F(x, y)$

Sujeto a:

$G(x, y) \leq 0$

$g(x, y) \leq 0$

En donde $x \in \mathbb{R}^{n_1}$ e $y \in \mathbb{R}^{n_2}$.

El valor óptimo del problema relajado es una cota inferior del valor óptimo del problema Bi-nivel. La solución óptima del problema relajado puede no ser factible en su correspondiente problema Bi-nivel original (Talbi, 2013) [20]. (Por más detalles de la relajación para el problema Bi-Nivel ver Estado del Arte).

Existen dos enfoques al problema Bi-Nivel, el enfoque optimista y el pesimista. El enfoque optimista los seguidores cooperan con el líder para que obtenga su mejor resultado, mientras que en el enfoque pesimista los seguidores buscan que el líder tenga el peor resultado (B. Colson et al., 2007) [18].

Complejidad de los problemas Bi-Nivel

Siendo generalmente no convexo y no diferenciable, los problemas Bi-Nivel son NP-Hard. En las instancias más “fáciles”, el problema de programación lineal Bi-Nivel, fue demostrado por R. G. Jeroslow (1985) [19] que son problemas NP-Hard. Ya que el conjunto inductivo no es convexo. Esto último se ve hasta en los casos más sencillos (donde todas las restricciones son lineales), al juntar los conjuntos de ambos niveles con sus respectivas restricciones, el conjunto resultante no es continuo E.G Talbi (2013) [20]. En efecto, muchos problemas de optimización combinatoria pueden reducirse a problemas Bi-Nivel. Por ejemplo, dado el siguiente problema de programación binario:

$min_{x,u}\ cx + eu$

Sujeto a:

$Ax + Eu \leq b$

$x \leq 0$

$u\ valor\ binario$

Donde $c \in \mathbb{R}^{n_x}, e \in \mathbb{R}^{n_u}, A \in \mathbb{R}^{m \times n_x}, E \in \mathbb{R}^{m \times n_u}, b \in \mathbb{R}^m$. *Notar que la variable binaria se pude representar alternativamente como* $0 = min\{u, 1-u\}$. *Por lo que se obtiene el siguiente problema Bi-Nivel:*

$min_{x,u,y}\ cx + eu$

Sujeto a:

$Ax + Eu \leq b$

$x \leq 0$

$y = 0$

$min_y \sum_{i=1}^{n_u} y_i$

Sujeto a:

$y \leq u$

$w \leq 1 - u$

En donde $y \in \mathbb{R}^{n_u}$. Se puede ver como un problema de programación binario NP-Hard se puede transformar en otro problema Bi-Nivel el cual también es NP-Hard.

Dificultad de resolución en problemas Bi-Nivel.

Comparado con los problemas de optimización de un nivel, la dificultad para resolver los problemas Bi-Nivel cae en los siguientes puntos:

- Evaluación de la solución en el nivel superior: No es fácil evaluar la función objetivo del nivel superior para problemas Bi-Nivel, ya que la función objetivo en el nivel superior no tiene una formulación explícita, ya que está compuesta por otro problema de optimización (problema de optimización de segundo nivel). Por lo tanto, la toma de decisiones del primer nivel no puede optimizarse sin tomar en cuenta la reacción (toma de decisiones) del problema de segundo nivel.
- Compleja interacción entre los problemas de nivel superior e inferior: Las restricciones del segundo nivel se pueden ver como restricciones no lineales, entonces el problema de programación es intrínsecamente no convexo. Incluso si todas las restricciones y funciones objetivo del primer nivel y del segundo nivel fueran lineales, no sería continua ni convexa para el problema de nivel superior (E.G Talbi, 2013) [20].