TESIS

a ser presentada el día 11 de Febrero de 2011, en la

Universidad de La República, UdelaR

para obtener el título de

MAGISTER EN INGENIERÍA MATEMÁTICA

para

Ing. François DESPAUX

Instituto de Investigación : LPE-IMERL
Componentes universitarios :
UNIVERSIDAD DE LA REPÚBLICA, FACULTAD DE INGENIERÍA

Título de la tesis :

Optimización de una Red de Datos IP/MPLS
sobre SDH/DWDM usando Tabú Search.
Caso de estudio: Red de Datos de un Operador de Telefonía Nacional

a realizarse el 11 de Febrero de 2011, por el comité de examinadores

Dr. Ing. Franco	ROBLEDO	Director de Tesis
Dr. Gerardo	RUBINO	Co-Director de Tesis
Dr. Jorge	IGLESIAS	Presidente
MSc. Ing. Maria	URQUHART	
MSc. Ing. Omar	VIERA	
MSc. Ing. Pedro	PIÑEYRO	
Ing. Diego	VALLE LISBOA	

Índice general

Agradecimientos

A mi tutor de tesis Dr. Ing. Franco Robledo por su permanente apoyo en aspectos técnicos y por su motivación constante, evaluando y aconsejando incesantemente de modo de poder perfeccionar a cada momento el desarrollo del trabajo. Al Dr. Gerardo Rubino, co-tutor de tesis por su aporte y apoyo a distancia.

Al equipo de trabajo por haberme permitido formar parte de este proyecto tan importante el cual permitió intercambiar conocimiento entre personas de primer nivel. En particular, al Ing. Andrés Corez por su apoyo en la implementación y programación de los algoritmos. A la Ing. Cecilia Parodi en el análisis y modelado de las entidades del problema y análisis de resultados. Al Dr.Ing. Eduardo Canale por su aporte en teoría de grafos y su constante instrucción en aspectos matemáticos.

A SCAPA matemática por haberme brindado la posibilidad de poder realizar esta Maestría en Ingeniería Matemática y por su constante apoyo siempre que fue necesario.

A mi familia, a mis amigos en general y todos aquellos que estuvieron presentes durante el desarrollo de trabajo y que de algún modo aportaron lo suyo para que este trabajo pudiera salir adelante. En particular, al MSc. Darío Padula por su apoyo incondicional y por su motivación constante, por permitirme parte de su tiempo para crear un ámbito de intercambio de ideas para poder clarificar aspectos del problema a resolver.

Por último agradecer al Dr. Jorge Iglesias, MSc. Ing. María Urquhart, MSc. Ing. Omar Viera, MSc. Ing. Pedro Piñeyro e Ing. Diego Valle Lisboa por el honor de evaluar este trabajo.

Abstract

El objetivo de esta tesis es presentar el diseño e implementación de una solución mediante un enfoque metaheurístico basado en Tabu Search para la resolución de un problema de diseñar una Red de Datos MPLS a ser desplegada sobre infraestructura de Transporte existente que a su vez es una combinación de tecnologías. Este trabajo da una aproximación para resolver el problema de como diseñar una red haciendo uso de recursos provistos por la infraestructura de transporte subyacente en forma robusta, esto es, que una falla simple (caída de un link) en la capa inferior no deje inoperativo ningún servicio, y cuyo costo sea óptimo.
En términos de complejidad computacional, el problema resuelto pertenece a la clase NP-HARD. Por tal motivo se ha buscado una solución aproximada de buena calidad dada la complejidad que existe para encontrar una solución exacta a esta clase de problemas.
Este trabajo fue desarrollado en el contexto de un proyecto mayor con ANTEL denominado “Optimización Bajo Diseño Robusto en Redes Multi-Overlay”, actividad 10 del Convenio Marco ANTEL-FING.

Parte I

Introducción

Capítulo 1

Introducción

1.. Motivación

En este trabajo se presentará la resolución a un problema de diseño de una Red de Datos MPLS a ser desplegada sobre infraestructura de transporte subyacente que a su vez es una combinación de tecnologías. Durante el desarrollo del trabajo y la ejecución de pruebas se estará trabajando sobre datos reales brindados por ANTEL, dado que como se explicó anteriormente, la tesis se encuentra enmarcada en el seno de un convenio ANTEL-FING.
Diferentes razones motivan el estudio de como las redes brindan servicios entre si, entre ellos imposición regulatoria, justificación económica, conveniencia organizacional, etc.

Históricamente, dada la complejidad del problema, los despliegues se centraban más que nada en torno a la factibilidad de la solución y no tanto a la optimalidad. En un escenario competitivo como el actual se hace imprescindible la búsqueda de soluciones que logren optimizar los costos más allá de la factibilidad de estas.

Dada la complejidad de este tipo de problemas y la dificultad para resolver el problema en forma exacta se optó por buscar una resolución aproximada basada en un enfoque metaheurístico. La solución implementada parte de una red inicial recibida como parámetro, por lo general sobredimensionada en la cantidad de posibles aristas de datos. Con el objetivo de optimizar dicha red se implementaron un conjunto de algoritmos, algunos basados en resultados teóricos estudiados durante el trabajo. Estos algoritmos son aplicados a los datos recibidos como input y como etapa final de la búsqueda de una solución óptima, se aplicaron los conceptos de la metaheurística *Tabu Search*.

2.. Objetivos

La solución hallada debe poder cursar cierto tráfico conocido cumpliendo con determinados parámetros de calidad. Además debe ser robusto ante fallas simples en la red de transporte subyacente, esto es, la caída de un link en la capa de transporte no debe inhabilitar los servicios provistos por la red de datos. Toda demanda afectada por dicha caída debe ser enrutada mediante otro camino que no se vea interrumpido por la falla.
Lo que se busca en este trabajo es dar respuestas a preguntas tales como

- En que centrales deberán instalarse nodos de la red MPLS.
- Entre que nodos de la red de datos conviene establecer enlaces y de que capacidades.
- En base a lo anterior, determinar que tecnología de transporte utilizar, determinar el camino en transporte a seguir para cada una de las conexiones obtenidas anteriormente.
- Cual es la forma óptima de distribuir el tráfico en la red MPLS ante cada uno de los escenarios de falla.

Los datos del problema son

- Conjunto de estaciones de red
- Matriz de tráfico correspondiente.
- Puntos donde pueden instalarse switches MPLS.
- Enlaces viables entre nodos.
- Velocidades soportadas para estos enlaces.
- Costos para cada alternativa de velocidad.
- Topología física de la red de transporte.

En síntesis, en esta tesis nos concentraremos en el modelado y resolución de un problema de diseño de una Red de Datos IP/MPLS sobre la infraestructura de transporte de ANTEL. Cuando modelamos formalmente el problema notamos que sus características diferían sustancialmente de otros modelos similares encontrados durante el estudio del estado del arte del diseño topológico de Redes Multi-Overlay. Realizando un análisis bibliográfico de la literatura existente en lo que tiene que ver con el diseño sobre Redes de Múltiples Capas, no hemos encontrado finalmente un modelo de optimización matemático que contemple las mismas restricciones que el modelo que presentamos aquí; modelo con especificidades muy propias de nuestro operador de telefonía nacional, ANTEL.

Cabe destacar que durante 2008 trabajamos intensamente con el Dr. Maurice Queyranne de la Universidad de British Columbia, quien realizó una visita al equipo de proyecto durante una semana. Su presencia fue fundamental para refinar detalles en el modelo de programación entera no lineal asociado al problema. Dicho modelo fue tomado posteriormente por la Ing. Cecilia Parodi en el contexto de su tesis de Maestría.

3.. Trabajos Previos

Como parte del trabajo de investigación se realizó una extensa búsqueda en la literatura existente de problemas que estén relacionados con el que aquí se abordará. Algunos problemas de modelos de optimización multi-overlay y diseño robusto de redes pueden verse en [1, 14, 17, 28].

En términos generales, cuando se diseñan problemas de redes suele haber un intercambio de productos o bienes entre diferentes nodos. En este proyecto, hay muchas demandas (tráfico) que fluye entre los distintos nodos que forman la Red de Datos. Problemas similares en los cuales hay un múltiple intercambio de productos o datos entre nodos fueron estudiados en [1, 3, 9, 16, 18].

Otro rasgo importante que debemos resolver en este trabajo es la factibilidad de las soluciones aún frente a un escenario de falla simple en transporte. Existen muchas técnicas para resolver este inconveniente tales como protección o recuperación; algunos ejemplos de esto pueden encontrarse en [23–25]. Otros trabajos en los que se estudió la supervivencia de redes de modo extensivo son [1, 11, 13].

Una amplia gama de trabajos sobre diseño óptimo de redes existe en la literatura. Algunas publicaciones pueden encontrarse en [12, 27, 28]. La primera considera únicamente el diseño de redes mientras que las dos restantes también tienen en cuenta la robustez ante fallas.

Otra de las características de nuestro problema a abordar es que el flujo entre nodos es indivisible; el mismo debe ser enrutado de manera integral. Mucho de los trabajos de diseño de redes estudiados consideran el flujo como divisible tal como puede verse en [1, 3, 4, 11, 13, 15, 16, 18, 22]. En [1, 3, 11, 15, 16, 22] se trabaja sobre redes de una capa mientras que el resto considera múltiples capas. Problemas sobre flujos indivisibles se referencian en [2, 26] pero una vez más, solo consideran el diseño de red en una capa.

4.. Organización del Documento

El presente documento se estructura de la siguiente manera:

Capítulo 1, Introducción, presenta una breve introducción y motivación al problema que se resolvió.

Capítulo 2, Descripción del Problema, explica brevemente algunos conceptos básicos de Redes y una descripción en alto nivel del problema resuelto.

Capítulo 3, Formalización de Entidades del Problema, describe como se modeló en alto nivel cada uno de los componentes de las redes, sus interconexiones, costo y otras variables.

Capítulo 4, Modelización Algebraica del Problema, formalización como problema de optimización combinatoria.

Capítulo 5, Complejidad Computacional, se presenta un análisis de Complejidad del problema.

Capítulo 6, Introducción - Tabú Search, se explican detalles y aspectos teóricos estudiados para la resolución del problema.

Capítulo 7, Heurística RMORN, muestra aspectos principales de la heurística implementada para resolver el problema RMORN.

Capítulo 8, Customización del Tabú Search, explica en detalle la customización de la metaheurística Tabú Search.

Capítulo 9, Descripción de los Casos de Prueba, presenta los casos de prueba utilizados para el testing de la implementación.

Capítulo 10, Análisis de Resultados , donde se muestran los resultados obtenidos al correr los casos de prueba.

Capítulo 11, Conclusiones , de la solución implementada.

Parte II

Marco Teórico

Capítulo 2

Descripción del Problema

1.. Introducción

A continuación se presentarán algunos conceptos de redes que son ampliamente utilizados a lo largo del documento y que es conveniente tenerlos presente a la hora de leer este trabajo.

1..1. Redes Overlay

Una red overlay es aquella que virtualiza sus conexiones sobre una o más redes existentes. En otras palabras las conexiones en una red overlay no son físicas sino lógicas. La Figura 2.1 muestra una representación de una Red Overlay. Lo que en el ejemplo se puede ver como un enlace directo

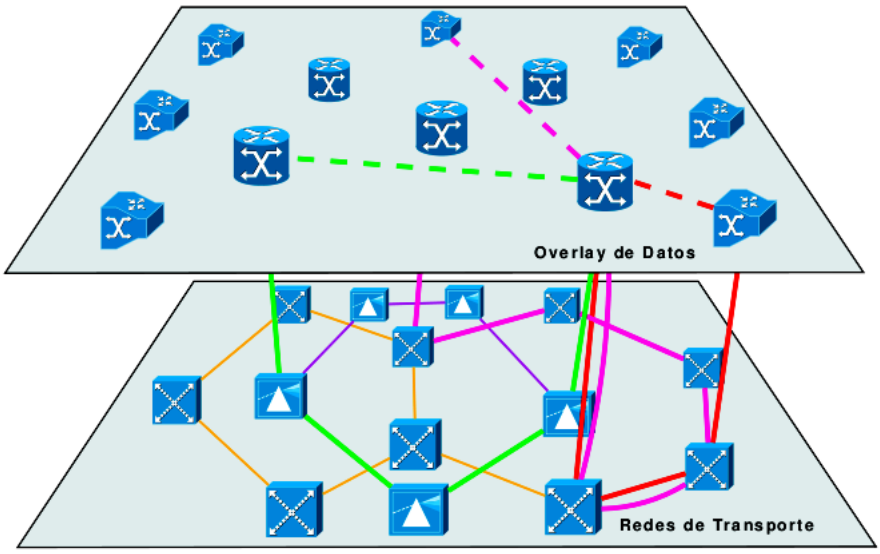

Figura 2.1: Ejemplo de Red Overlay.

entre dos nodos en la red de arriba es en realidad una conexión punto a punto provista por la red de transporte. Ejemplos típicos de estas redes son **Internet**, **Redes P2P**, **Redes IPSec**, **Redes ATM** y

Redes MPLS.

1..2. MPLS

MPLS (Multi-Protocol Label Switching) es una red privada IP que combina la flexibilidad de las comunicaciones punto a punto o Internet y la fiabilidad, calidad y seguridad de los servicios Private Line, Frame Relay o ATM.
Ofrece niveles de rendimiento diferenciados y priorización del tráfico, así como aplicaciones de voz y multimedia. Y todo ello en una única red.
Algunas características de las redes MPLS son

- MPLS Intenta conseguir las ventajas de ATM, pero sin sus inconvenientes.
- Asigna a los datagramas de cada flujo una etiqueta única que permite una conmutación rápida en los routers intermedios (solo se mira la etiqueta, no la dirección de destino).
- MPLS se basa en el etiquetado de los paquetes en base a criterios de prioridad y/o calidad (QoS).
- La idea de MPLS es realizar la conmutación de los paquetes o datagramas en función de las etiquetas añadidas en capa 2 y etiquetar dichos paquetes según la clasificación establecida por la QoS en la SLA.
- Por tanto MPLS es una tecnología que permite ofrecer QoS, independientemente de la red sobre la que se implemente.
- El etiquetado en capa 2 permite ofrecer servicio multiprotocolo y ser portable sobre multitud de tecnologías de capa de enlace: ATM, Frame Relay, líneas dedicadas, LANs.

1..3. Redes de Acceso

Se denomina red de acceso al trayecto final de las redes de telecomunicaciones, o sea, el tramo que une el domicilio del usuario con el resto de la red.
Saber el punto que determina exáctamente donde comienza la red de acceso no es algo fácil de establecer y generalmente depende del diseño exacto de la red y de la o las tecnologías involucradas.
El tráfico de usuarios individuales accede a la red de acceso y es esta red la que encaminará el tráfico hacia la red de transporte.

1..4. Redes de Transporte

El objetivo de estas redes es concentrar el tráfico de información proveniente de las redes de acceso para llevarlo a mayores distancias. Tradicionalmente sus características y arquitectura dependían del tipo de información que se deseaba transportar y a las características de las redes de acceso. De este modo, existen redes de transporte de señal de televisión, redes de transporte de televisión por cable, redes de transporte de telefonía fija y de comunicaciones móviles, entre otras. Con el advenimiento

de la digitalización comenzó un proceso de convergencia en las redes de transporte para hacerlas capaces de transportar cualquier tipo de información. Un aporte significativo a este proceso fue el uso masivo de fibra óptica como medio físico de preferencia para estas redes. Tecnologías digitales tales como X25, Frame Relay, SDH y ATM han ido surgiendo a lo largo de los años.

1..5. Red de Datos

Son aquellas infraestructuras o redes de comunicación que han sido diseñadas especificamente para la transmisión de información mediante el intercambio de datos.
Generalmente están basadas en conmutación de paquetes[1] y se clasifican de acuerdo a su tamaño, distancia que cubren y su arquitectura física. Algunas clases de redes de datos son las LAN, WAN, MAN.

2.. El Problema a Abordar

A continuación y luego de una brevísima introducción a los principales conceptos sobre redes que se utilizan ampliamente a lo largo de este trabajo se dará una introducción en alto nivel del problema a resolver. En el siguiente capítulo se dará una descripción más detallada y formal de cada uno de los componentes del problema, de su notación y de como fueron modelados.

El problema consiste en el diseño de una Red de Datos multi-overlay que será desplegada sobre una red de transporte existente. Sobre esta red de datos se pretende entregar servicios de distinto tipo y por ende con distintos requerimientos. Se buscará minimizar los costos incurridos por el despliegue de la infraestructura de transporte.
Como datos de entrada se cuenta con una Red de Transporte basada en la actual estructura de transporte de ANTEL que en términos topológicos es una red anillada y planar[2], particularidad interesante que permitió aplicar resultados teóricos de teoría de grafos. Sobre la Red de Transporte se estará montando la Red de Datos a diseñar. El transporte soporta diferentes tecnologías y determinar con qué se dimensionará cada link de esta red es parte del problema a resolver. Esta red cuenta además con un conjunto de estaciones de red, típicamente centrales telefónicas. Como parte de los datos de entrada o input se cuenta con un conjunto de links de datos que serán los candidatos a formar parte de la red. Además se cuenta con un conjunto de nodos MPLS donde cada uno estará asociado con algún nodo de la Red de Transporte. También es conocida la matriz de demanda entre distintos puntos, demanda que está determinada por cada uno de los servicios que se quieren desplegar sobre la red. La idea entonces será construir la Red de Datos de modo de poder satisfacer cada una de las demandas especificadas en la matriz de demanda tratando de que esta construcción sea óptima en términos de costos, lo que equivale a decir que se haga un uso eficiente de la red de transporte, buscando encontrar enrutamientos óptimos entre los puntos en cuestión y teniendo en cuenta además que la solución debe ser robusta ante fallas simples en el transporte; esto es, que ante la caída de un link físico las demandas afectadas deben poder ser enrutadas buscando un camino alternativo. El algoritmo entonces busca una solución que sea factible tanto para el escenario normal (sin fallas) como para cada uno de los escenarios de falla simple de links en transporte.

[1] Conmutación de Paquetes (Packet Switching) es un intercambio de bloques de información o paquetes con un tamaño específico entre dos puntos, un emisor y un receptor

[2] En teoría de grafos, un grafo es plano si puede ser dibujado en el plano sin que ninguna arista se cruce. Si modelamos nuestra Red de Transporte como un grafo entonces podremos verlo como un grafo planar.

Un análisis que se hará también durante la búsqueda de la solución está asociado con el dimensionamiento de los links. Es input del problema también las diferentes tecnologías que podrán utilizarse y sus respectivos costos. Por tanto, determinar que tecnología utilizar en un link particular junto con el dimensionamiento son parte del problema a resolver.

Capítulo 3

Formalización de Entidades del Problema

En este capítulo se definirán de manera formal las principales entidades y características relevantes del problema.

El problema consta de dos Redes, la Red Datos y la Red de Transporte. En primer lugar se hará una descripción de la Red de Datos y sus componentes, posteriormente se hará un análisis de la Red de Transporte y por último se explicarán las relaciones entre ambas redes.

1.. Red de Datos

La Red de Datos se representará mediante un grafo $G_D = (V_D, E_D)$ donde V_D representa el conjunto de nodos de la red y E_D representa las aristas o conexiones entre nodos.
El grafo G_D es un grafo no dirigido. Dada las características de densidad y mallado que suele tener esta red es probable que el grafo no sea planar. Otro punto destacable de esta red respecto a la de transporte es la naturaleza de su tráfico. En la Red de Datos el tráfico es dinámico, tanto las rutas como el volumen pueden cambiar con el tiempo.

1..1. Nodos de Datos

Los nodos de la Red de Datos pueden clasificarse en dos clases, los nodos Access, representados como V_A y los nodos Edges, representados como V_E. Los primeros hacen las veces de concentradores de tráfico de un conjunto de clientes y alimentan la red conectándose físicamente a uno o más nodos Edges.
Ejemplo típico de este tipo de nodos son los DSLAM xDSL y se representarán gráficamente como se muestra en la figura 3.1

Los nodos Edges en cambio poseen mayor inteligencia ya que son capaces de incorporar mecanismo de Ingeniería de Tráfico que les permite elegir adecuadamente rutas viables, no congestionadas y libres de fallas dado el conocimiento que cada uno de estos nodos tiene respecto a la topología de

Figura 3.1: Nodo Access

la red. Tienen la capacidad de poder enrutar el tráfico hacia su destino.
Estos nodos se representarán gráficamente de la siguiente manera

Figura 3.2: Nodo Edge

Estos nodos Edges se conectan tanto a nodos de Acceso como a otros nodos edges. En particular, es de interés para este trabajo la subred generada por el conjunto de nodos Edges la cual llamaremos $G_E = (V_E, E_E)$ ya que los algoritmos implementados tomarán la topología de la misma para determinar el ruteo y asignación de capacidades de sus aristas. La relación entre ambas clases de nodos es la siguiente: $V_A \cup V_E = V_D$ y a su vez $V_A \cap V_E = \emptyset$

Una segunda distinción entre nodos de datos es aquella que clasifica a los nodos según sean fijos en la solución u opcionales. De este modo se definen los nodos fijos como V_F y los nodos opcionales o de *Steiner* como V_S. Los nodos fijos son aquellos que deben estar presentes en la solución, por ejemplo, aquellos nodos que enviarán o recibirán tráfico. Los nodos de *Steiner* no necesariamente formarán parte de la solución y estarán determinados por los resultados que surjan de correr los algoritmos de ruteo. Los nodos de acceso V_A también forman parte de los nodos fijos y por lo tanto se cumple que $V_A \subseteq V_F$.
En lo que respecta al tráfico que deberá cursarse a lo largo de la red de datos, se definen dos tipos:

- Tráfico Comprometido: El cual deberá ser enrutado aún en escenarios simples de falla, esto es, si determinado tráfico es enrutado por una ruta r y una falla ocurre en alguna arista intermedia entonces dicho tráfico deberá ser enrutado por otra ruta r' que no contenga a la arista que falló. Típicamente es tráfico asociado a servicios de VoIp[1] ó VoD[2].
- Tráfico Eventual: Se acepta que este tráfico, típicamente tráfico de Internet, esté disponible un cierto porcentaje de tiempo.

Se define entonces, para cualesquiera $v_i, v_j \in V_D$ el vector de tráfico entre estos nodos como $\vec{m}_{ij} = (\dot{m}_{ij}, \ddot{m}_{ij})$ donde $\dot{m}_{ij}$ representa el tráfico comprometido y $\ddot{m}_{ij}$ el tráfico eventual.
Estamos en condiciones entonces de definir la matriz de demanda del problema.

Definición 3.1. *Sea $V_D = \{v_1, v_2, ..., v_h\}$, existe una matriz $\bar{M} = ((\vec{m}_{ij}))_{1 \leq i,j \leq h}$ llamada matriz de demanda donde los vectores $\vec{m}_{ij} = (\dot{m}_{ij}, \ddot{m}_{ij}) \in \{\mathbb{R}_0^+ \times \mathbb{R}_0^+, \forall v_i, v_j \in V_D\}$. $\dot{m}_{ij}$ es el tráfico comprometido entre los nodos v_i y v_j mientras que $\ddot{m}_{ij}$ es el tráfico eventual entre ellos.*

[1] Voice over IP protocol
[2] Video on Demand

De acuerdo a lo expresado anteriormente, debe cumplirse que si $\vec{m}_{ij} \neq 0 \Rightarrow v_i, v_j \in V_F$

1..2. Las Aristas de Datos

Cada link de datos $e_{ij} = (v_i, v_j) \in E_D$ representa una potencial arista de datos a ser considerada como parte de la solución del problema. Una característica de estos links de datos es su capacidad la cual estará comprendida en el conjunto definido por $\hat{B} = \{\hat{b}_0, \hat{b}_1, ..., \hat{b}_{\bar{B}}\}$. El conjunto de capacidades permitidas estará determinada por las tecnologías aplicables a la red de transporte y son inputs para nuestro problema. Cada arista de datos será dimensionada con algún valor del conjunto $\hat{B}$ y se asume que si un link no forma parte de la solución este será dimensionado con una capacidad igual a cero. Por esto último, se define la capacidad $\hat{b}_0 = 0$. Queda entonces determinado el conjunto de aristas que componen la solución como aquellas aristas cuya capacidad es distinta de cero. Además, es posible saber si un nodo de datos v_i pertenece o no a la solución si existe algún v_j tal que $b_{ij} \neq 0$, donde b_{ij} representa la capacidad con que fue dimensionado la arista (i, j).
Dado que el grafo es no dirigido se cumple que $b_{ij} = b_{ji}$.
Un ejemplo de la Red de Datos y la interacción entre los nodos edges, nodos access y los links de datos se muestra en la siguiente figura.

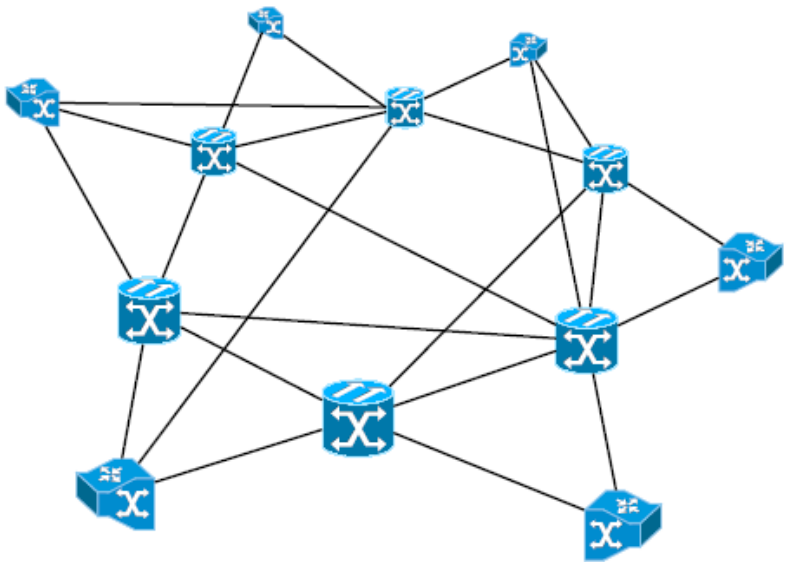

Figura 3.3: Red de Datos

2.. Red de Transporte

Al igual que la Red de Datos, la Red de Transporte también se modelará como un grafo no dirigido $G_T = (V_T, E_T)$ donde V_T representa el conjunto de terminales o centrales y E_T los links físicos entre nodos. Esta red, en términos topológicos es una red anillada y su grafo asociado G_T tiene la particularidad de ser 2-arista conexo. Además es una red planar, propiedad que nos permitirá aplicar ciertos resultados teóricos como se describirá más adelante.
En lo que respecta al tráfico, a diferencia de la Red de Datos, este es completamente estático. Esta-

blecida una demanda entre dos puntos, hay que buscar un recorrido en la red con capacidad suficiente, y una vez establecido éste, el ancho de banda requerido se descuenta de los enlaces involucrados.

2..1. Nodos de Transporte

Definición 3.2. *Llamaremos Estación de Red al edificio donde está físicamente instalada la infraestructura de telecomunicaciones. Son los edificios donde convergen las canalizaciones de la planta de cobre y de fibra óptica y donde residen físicamente nodos de transporte y de datos entre otros componentes.*

Como simplificación al problema supondremos que en cada Estación de Red existe un único nodo de transmisión. El nodo de transporte será representado gráficamente como se muestra a continuación

Figura 3.4: Nodo de Transporte

2..2. Aristas de Transporte

Físicamente las aristas de transporte se reducen a las canalizaciones que conectan las Estaciones de Red. Se asumirá que la capacidad de estas aristas es infinito o lo que es equivalente que deberán instalarse tantos equipos de transporte como sean necesarios para poder cumplir con las demandas de conectividad de la Red de Datos. En nuestro modelo, la falla en un tramo afecta a todas las conexiones que pasan por esta arista.

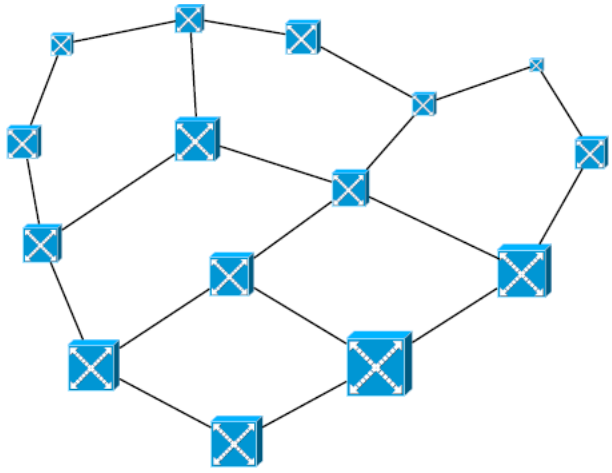

Figura 3.5: Red de Transporte

2..3. Relación entre Transporte y Datos

En una Estación de Red siempre existirá un único Nodo de Transporte. Sin embargo, no ocurre lo mismo con los nodos de datos los cuales podrán estar presentes o no. En caso de estar presentes podrán haber a lo sumo dos nodos de datos. Para el caso en que hayan dos nodos estos serán necesariamente de distinto tipo, un nodo de acceso y el otro un nodo edge.
Se define la función *Network Station* como

$$ns : (V_D \cup V_T) \times (V_D \cup V_T) \to \{0, 1\} \tag{3.1}$$

en la cual se cumple que $ns(u, v) = 1$ si y solo si u y v pertenecen a la misma estación.

Dado un nodo de datos queda determinado el correspondiente nodo de transporte o estación t mediante la función *tns* la cual se define como $tns : V_D \to V_T$ tal que $tns(v) = t$ cuando $ns(v, t) = 1$. Esto significa que el nodo v y la estación t se encuentran en el mismo lugar físico.

2..4. Atributos del Problema

Tráfico

Ya se describió anteriormente las características del tráfico que intercambian uno o más nodos entre si. También se vio una representación mediante una matriz de demanda que refleja para cada uno de los nodos como intercambian trafico con otros, tanto el eventual como el comprometido. Recordemos entonces que dicho tráfico se puede representar como $\vec{m}_{ij} = (\dot{m}_{ij}, \ddot{m}_{ij})$ donde $\dot{m}_{ij}$ representa el tráfico comprometido y $\ddot{m}_{ij}$ representa el tráfico eventual.

Costos

El objetivo de este trabajo es poder optimizar los costos del despliegue de una red multioverlay. La Red de Transporte juega un papel preponderante en lo que tiene que ver con los costos. En esta sección se analizará en detalle los costos que atañen al problema.

Empezaremos con una definición útil.

Definición 3.3. *ρ_T^{ij} es el conjunto de caminos en transporte que un flujo entre dos nodos de transporte $tns(v_i), tns(v_j) \in V_T$ seguirá en una solución al problema.*

A la hora de estudiar los costos en Transporte dos alternativas tecnológicas fueron tenidas en consideración:

- **Tecnología TDM**[3] El costo depende tanto de la distancia entre las estaciones así como también del ancho de banda seleccionado. Esta claro que cuanto más distantes estén los nodos entre si la excavación necesaria para tender la fibra será más costosa.

[3]Time Division Multiplexing

Resumiendo, el costo de transmisión de un flujo respecto a las tecnologías tradicionales (PDH/SDH)[4] tiene la expresión

$$cost(\rho_T^{ij}, b_{ij}) \quad = k \times r(\rho_T^{ij}) \times b_{ij} \tag{3.2}$$

donde k es una constante, $r(\rho_T^{ij})$ es la distancia del camino ρ_T^{ij} en la Red de Transporte y b_{ij} es la capacidad asignada a la arista (v_i, v_j) en la Red de Datos.

- **Tecnología DWDM**[5] El costo depende solo de la distancia. Esto ocurre debido al hecho de que en DWDM, una longitud de onda particular es elegida acorde al ancho de banda requerido por el link. Muchas conexiones pueden establecerse sobre el mismo link físico utilizando diferentes valores de longitud de onda. Por tanto, el único punto relevante aquí es la distancia:

$$cost(\rho_T^{ij}, b_{ij}) \quad = k \times r(\rho_T^{ij}) \tag{3.3}$$

Cada una de las alternativas tecnológicas tiene sus ventajas dependiendo de la capacidad a instalar. En el gráfico de la Figura 3.6 puede verse que de acuerdo a la capacidad necesitada será conveniente hacer uso de una tecnología particular o la otra. Si la capacidad requerida es pequeña entonces SDH es conveniente. En cambio, si se necesita un link con alta capacidad entonces DWDM es la elección.

El costo es proporcional a la distancia. Se asume que cada una de las arista de transporte den-

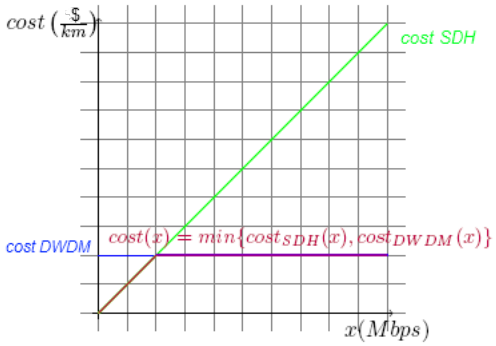

Figura 3.6: Función de Costos

tro de un camino tienen asignada la misma capacidad.

La opción menos cara a elegir depende del ancho de banda necesitado. En consecuencia, se adopta el formalismo de una función T que devuelve el mínimo costo por kilómetro de una tecnología en particular de acuerdo al ancho de banda requerido.
Por tanto, el costo es calculado como :

$$cost(\rho_T^{ij}, b_{ij}) \quad = r(\rho_T^{ij}) \times T(b_{ij}) \tag{3.4}$$

[4] Plesiochronous/Synchronous Digital Hierarchy
[5] Dense Wavelength Division Multiplexing

Ruteo del Tráfico

Sea una matriz de demanda $\bar{M}$. El diseño de la red debe responder explicitamente a como el tráfico con origen v_i y destino v_j y $\vec{m}_{ij} \neq \vec{0}$ se propaga a lo largo de la red. De modo que existe un conjunto de links de datos que llamaremos *túnel* que representa la ruta fija que el tráfico seguirá desde el origen v_i al destino v_j.

Enmarcaremos al conjunto de todos los túneles de G_D dentro del conjunto P_D y diremos además que un escenario de ruteo ρ_D es un conjunto de caminos en G_D o bien un subconjunto de P_D.

Ejemplo.
Sea la Red de Datos de la figura 3.7 con el siguiente conjunto de nodos y de aristas:
$V_D = \{v_1, v_2, v_3, v_4, v_5, v_6\}$
$E_D = \{(v_1, v_2), (v_1, v_3), (v_1, v_6), (v_2, v_3), (v_2, v_5), (v_2, v_6), (v_3, v_4), (v_3, v_5), (v_4, v_5), (v_5, v_6), \}$

Para el escenario de ruteo:
$\rho_D = \{\{(v_1, v_2)\}, \{(v_1, v_3)\}, \{(v_1, v_2), (v_2, v_3)\}, \{(v_2, v_3)\}, \{(v_2, v_6)\}, \{(v_2, v_3), (v_3, v_5), (v_5, v_6)\}$
$, \{(v_3, v_4), (v_4, v_5)\}, \{(v_4, v_5)\}, \{(v_5, v_6), (v_6, v_1)\}.$

el conjunto de rutas desde el nodo v_1 al nodo v_3 son $\{(v_1, v_3)\}$ y $\{(v_1, v_2), (v_2, v_3)\}$.

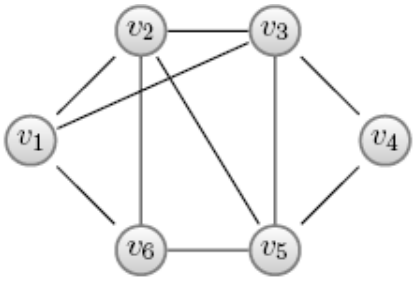

Figura 3.7: Ejemplo de Ruteo

Robustez

Uno de los aspectos que tornan el problema un poco más complejo es el requerimiento de robustez de la red a diseñar ante fallas simples en la Red de Transporte.
Usualmente, las fallas en la Red de Transporte ocurren cuando la fibra óptica se rompe debido por ejemplo a maquinaria encargada de realizar mantenimiento o nuevos dragados que involuntariamente rompe una o más fibras. Esta ruptura se traduce en la caída de un link de transporte que usualmente implicará la caída de uno o más aristas en la Red de Datos. Como resultado, todos los túneles que contenga a estas aristas de datos afectadas se verán afectados y por ende todas aquellas demandas que son enrutadas por estos túneles.
En el ejemplo de la figura 3.8 podemos ver gráficamente como sería el caso de una falla simple en transporte. En este caso, si la arista $e \in E_T$ falla los links de datos $u - v$ y $r - s$ que están mapeados a caminos en transporte que utilizan la arista e fallarán también. Como resultado, todos los túneles

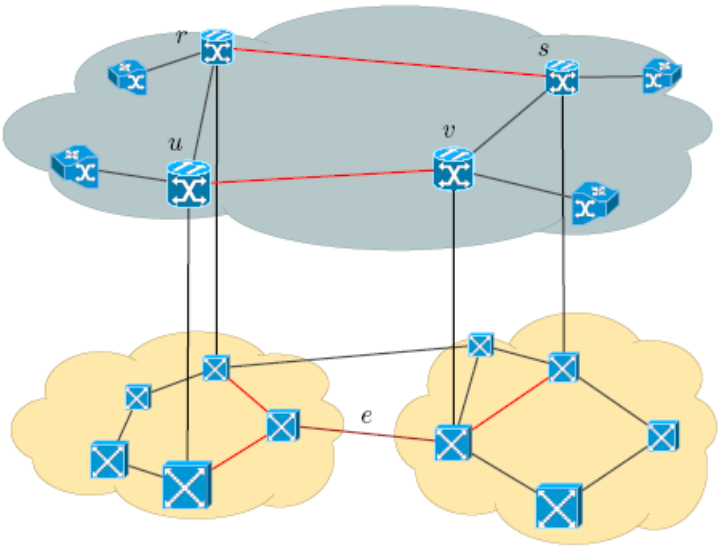

Figura 3.8: Falla en Transporte

que utilicen estas aristas de datos se verán afectados también.

Aún en un escenario de falla simple todas las demandas de tráfico deberán ser enrutadas. Esto implica que ante un escenario de falla como el anterior, deberá buscarse un camino alternativo para las demandas afectadas, esto es, túneles que permitan encausar el tráfico afectado y que no utilicen las aristas $u - v$ y $r - s$ y por tanto, que no utilicen la arista e en la Red de Transporte subyacente.

En este capítulo se definieron de manera descriptiva entidades y conceptos que forman parte del problema. En el siguiente capítulo se definirán formalmente las entidades, relaciones y restricciones y se llegará a la formulación del problema como un problema de optimización combinatoria.

Capítulo 4

Modelización Algebraica del Problema

1.. Modelo Completo

En esta sección, se adopta un enfoque matemático y muchos conceptos ya presentados se formalizan en definiciones.

1..1. Definiciones

Definición 4.1. *$G_D = (V_D, E_D)$ y $G_T = (V_T, E_T)$ son dos grafos no dirigidos que modelan la Red de Datos y la Red de Transporte respectivamente.*

Definición 4.2. *Sea $G_D = (V_D, E_D)$ la Red de Datos, existen dos clasificaciones de los nodos que la componen:*

- *Nodos de Acceso V_A y nodos Edges V_E que forman una partición del conjunto de nodos de datos, es decir, $V_D = V_A \cup V_E$ y $V_A \cap V_E = \emptyset$.*
- *Nodos fijos V_F y nodos* Steiner *V_S que forman una partición del conjunto de nodos de datos, es decir, $V_D = V_F \cup V_S$ y $V_F \cap V_S = \emptyset$.*

Nota 4.1. $V_A \subset V_F$

Definición 4.3. *Sea $V_D = \{v_1, v_2, ..., v_h\}$ existe una matriz de tráfico $\bar{M} = ((\vec{m}_{ij}))_{1 \leq i,j \leq h}$ donde los vectores $\vec{m}_{ij} = (\dot{m}_{ij}, \ddot{m}_{ij}) \in \{\mathbb{R}_0^+ \times \mathbb{R}_0^+, \forall v_i, v_j \in V_D\}$. $\dot{m}_{ij}$ es el tráfico comprometido entre los nodos v_i y v_j mientras que $\ddot{m}_{ij}$ es el tráfico eventual entre ellos.*

Nota 4.2. *Para ser coherente con la realidad se asume que la matriz $\bar{M}$ es simétrica.*

Definición 4.4. *Sea $E_D = \{(v_i, v_j) : 1 \leq i, j \leq h\}$ el conjunto de links de datos. Entonces $\hat{B} = \{\hat{b}_0, \hat{b}_1, ...\hat{b}_{\bar{B}}\}$ donde $0 = \hat{b}_0 < \hat{b}_1 < ... < \hat{b}_{\bar{B}}$ es el conjunto posible de capacidades para dimensionar los links de datos.*

El problema de escoger las capacidades de los links es formulada como la búsqueda de una función $b : E_D \to \hat{B}$ donde denotamos $b(v_i, v_j) = b_{ij} \in \hat{B}$.

Como se dijo en el Capítulo 3, en cada estación de red existe siempre un nodo de transporte y potencialmente puede encontrase también algún nodo de datos. Esto inspira la siguiente definición:

Definición 4.5. *Sea* $ns : (V_D \cup V_T) \times (V_D \cup V_T) \rightarrow \{0, 1\}$ *la función de estación de red. Se cumple que* $ns(u, v) = 1 \Leftrightarrow u$ *y* v *pertencen a la misma estación.*

Es claro que $ns(u, v) = 0\ \forall u, v \in V_T$ y $\forall v \in V_D, \exists t \in V_T / ns(v, t) = 1$.

Definición 4.6. *Sea* $tns : V_D \rightarrow V_T$ *la función estación de red de transporte.* $tns(v) = t$ *cuando* $ns(v, t) = 1$.

Definición 4.7. *Sea* $G_D = (V_D, E_D)$*,* P_D *es el conjunto de todos los posibles caminos en el grafo* G_D*. Definimos escenario de ruteo a cada subconjunto* $\rho_D \in P_D$ *y* $g_D : V_D \times V_D \rightarrow 2^{P_D}$ *es la función de rutas que determina todos los posibles caminos entre dos pares de nodos de datos.*

Análogamente,

Definición 4.8. *Sea* $G_T = (V_T, E_T)$*,* P_T *es el conjunto de todos los posibles caminos en el grafo* G_T*. Definimos configuración de flujo a cada subconjunto* $\rho_T \in P_T$ *y* $g_T : V_T \times V_T \rightarrow 2^{P_T}$ *es la función de flujos que determina todos los posibles caminos entre dos nodos de transporte.*

Definición 4.9. *Sea* $G_D = (V_D, E_D)$*, un escenario de ruteo* $\rho_D \in P_D$ *y dos nodos de datos* v_i *y* $v_j \in V_D$*, las rutas desde* v_i *a* v_j *son los elementos del conjunto* $\rho_D^{ij} = g_D(v_i, v_j) \cap \rho_D$.

Definición 4.10. *Sea* $G_T = (V_T, E_T)$*,* $G_D(V_D, E_D)$*, una configuración de flujo* $\rho_T \in P_T$ *y un link* $e_d \in E_D$ *tal que* $e_d = (v_i, v_j), v_i, v_j \in V_D$*, la implementación del flujo para* e_d *son los elementos del conjunto* $\rho_T^{ij} = g_T(tns(v_i), tns(v_j)) \cap \rho_T$.

Definición 4.11. *Sean las redes* $G_D = (V_D, E_D)$ *y* $G_T = (V_T, E_T)$*, llamamos función de escenarios de ruteo a* $\Phi : (E_T \cup \emptyset) \rightarrow 2^{P_D}$ *tal que cada* $e_t \in E_T$ *y el conjunto vacío es asignado a un escenario de ruteo* $\rho_D \subset P_D$*. En particular,* $\Phi(\emptyset)$ *es denominado escenario de ruteo nominal.*

Definición 4.12. *Sean las redes* $G_D = (V_D, E_D)$ *y* $G_T = (V_T, E_T)$*, llamamos función de configuración de flujos a la función* $\Psi : E_D \rightarrow 2^{P_T}$ *tal que a cada* $e_d \in E_D$ *le es asignada una configuración de flujo* $\rho_T \subset P_T$.

Definición 4.13. *Sea* $T : \hat{B} \rightarrow \mathbb{R}$ *la función de costo por tecnología tal que asigna a cada capacidad disponible un costo por unidad de longitud. Se asume que* $T(\hat{b}_0) = T(0) = 0$.

Definición 4.14. *Sea* $r : E_T \rightarrow \mathbb{R}$ *la función distancia tal que asigna a cada arista de transporte su longitud. Es posible extender el dominio de dicha función a cada implementación de flujo* ρ_T^{ij}*, considerando la suma de las distancias de sus links, es decir,* $r : E_T^* \rightarrow \mathbb{R}$ *donde* $r(\rho_T^{ij}) = \sum_{e \in \rho_T^{ij}} r(e)$*. Se asume que* $r(\emptyset) = 0$.

Definición 4.15. *Se define cost :* $P_T \times \hat{B}$ *la función costo tal que* $cost(\rho_T^{ij}, b_{ij}) = r(\rho_T^{ij}) \times T(b_{ij})$ *retorna el costo incurrido en la red de transporte para un flujo cuyo origen es el nodo* v_i *y su destino el nodo* v_j*, que sigue el camino* ρ_T^{ij} *y tiene capacidad* b_{ij}.

Definición 4.16. *Sea* $z_Q : \mathbb{R} \rightarrow \mathbb{R}$ *la función de capacidad de tráfico eventual tal que asigna a cada ancho de banda* $\ddot{m}_{ij}$*, la mínima capacidad requerida para las aristas que llevan dicho tráfico.*

1..2. Modelo Matemático

Una vez definidas formalmente las entidades y objetos que representan la realidad del problema pasamos a definir las restricciones y relaciones de los mismos.

Primero identificamos los datos del problema:

Red de Datos

- $G_D = (V_D, E_D)$: Grafo que representa la Red de Datos junto con sus nodos y links.
- V_A: Conjunto de nodos de acceso.
- V_E: Conjunto de nodos de edges.
- $\bar{M}$: Matriz de Tráfico.
- z_Q: función de capacidad de tráfico eventual.

Red de Transporte

- $G_T = (V_T, E_T)$: Grafo que representa la Red de Transporte, nodos y links.
- $\hat{B}$: Capacidades disponibles.
- $T(\hat{b}_{ij})\ \forall \hat{b}_{ij} \in \hat{B}$: Costo de las capacidades por kilómetro.
- $r(e)\ \forall e \in E_T$: longitud de los links.
- ns: función estación de red.
- tns: función estación de red de transporte.

En segundo lugar, recordamos que las variables del problema son:

- **B**: capacidad asignada a cada link de datos.
- ρ_D: rutas en Red de Datos
- ρ_T: caminos en Red de Transporte.

En este punto, se tienen todos los elementos para proponer un modelo completo de optimización el cual será referido como **MORN** (Multi-Overlay Robust Network) y tiene la siguiente forma matemática:

$$
MORN \begin{cases}
\min\limits_{B,\Phi,\Psi} \sum\limits_{\substack{\rho_T=\Psi(e) \\ e=(v_i,v_j) \\ e\in E_D}} r(\rho_T^{ij})T(b_{ij}) & \\
|\rho_T^{ij}| = 1 & \forall e\in E_E, e=(v_i,v_j), b_{ij}\neq 0, \rho_T=\Psi(e). \\
|\rho_T^{ij}| = 2 & \forall e\in E_D, e=(v_i,v_j), v_i\in V_A, v_j\in V_E, b_{ij}\neq 0, ns(v_i,v_j)=0, \rho_T=\Psi(e). \\
\sum\limits_{\substack{b_{ij}\neq 0 \\ \forall v_j\in V_E}} 1 = 1 & \forall v_i\in V_A. \\
\sum\limits_{\substack{b_{ij}\neq 0 \\ \forall v_j\in V_A}} 1 = 0 & \forall v_i\in V_A. \\
b_{pq} \geq \sum\limits_{\forall v_j\in V_D} \dot{m}_{pj} + z_Q \left(\sum\limits_{\forall v_j\in V_D} \ddot{m}_{pj} \right) & \forall v_p\in V_A, v_q\in V_E, b_{pq}\neq 0. \\
\vec{m'}_{ij}^{\,\emptyset} = \vec{m}_{ij} + \sum\limits_{\substack{b_{ei}\neq 0 \\ \forall v_e\in V_A}} \vec{m}_{ej} + \sum\limits_{\substack{b_{fj}\neq 0 \\ \forall v_f\in V_A}} \vec{m}_{if} + \sum\limits_{\substack{b_{ei}\neq 0 \\ \forall v_e\in V_A}} \sum\limits_{\substack{b_{fj}\neq 0 \\ \forall v_f\in V_A}} \vec{m}_{ef} & \forall v_i, v_j\in V_E. \\
\vec{m'}_{ij}^{\,t} = \vec{m}_{ij} + \sum\limits_{\substack{\Psi(ei)\cap\{t\}=\emptyset \\ b_{ei}\neq 0 \\ \forall v_e\in V_A}} \vec{m}_{ej} + \sum\limits_{\substack{\Psi(fj)\cap\{t\}=\emptyset \\ b_{fj}\neq 0 \\ \forall v_f\in V_A}} \vec{m}_{if} + \sum\limits_{\substack{\Psi(ei)\cap\{t\}=\emptyset \\ b_{ei}\neq 0 \\ \forall v_e\in V_A}} \sum\limits_{\substack{\Psi(fj)\cap\{t\}=\emptyset \\ b_{fj}\neq 0 \\ \forall v_f\in V_A}} \vec{m}_{ef} & \forall v_i, v_j\in V_E, \forall t\in E_T. \\
|\rho_{D_t}\cap t| = 0 & \forall t\in E_T, \rho_{D_t}=\Phi(t). \\
|\rho_{D_t}^{ij}| = 1 & \forall v_i, v_j\in V_E, \forall t\in\{\emptyset\cup E_T\}, \rho_{D_t}=\Phi(t), \vec{m'}_{ij}^{\,t}\neq\vec{0}. \\
b_{pq} \geq \sum\limits_{\forall v_i, v_j\in V_E} \left(\dot{m'}_{ij}^{\,t} |\rho_{D_t}^{ij}\cap e| \right) + z_Q \left(\sum\limits_{\forall v_i, v_j\in V_E} \left(\ddot{m'}_{ij}^{\,t} |\rho_{D_t}^{ij}\cap e| \right) \right) & \forall e\in E_E, e=(v_p,v_q), \forall t\in\{\emptyset\cup E_T\}, \rho_{D_t}=\Phi(t), \vec{m'}_{ij}^{\,t}=(\dot{m'}_{ij}^{\,t}, \ddot{m'}_{ij}^{\,t}).
\end{cases}
$$

A continuación se detalla cada una de las restricciones del modelo.

- El primer término es la función objetivo a optimizar. Consiste de la suma de los costos que debe minimizarse. Para este propósito, la terna de funciones (B, Φ, Ψ) debe ser obtenida.

  En cada término de la suma, se le suma el costo en transporte de cada link de datos $e \in E_D$ incluído en la solución. Como se explicó previamente, dicho costo es expresado como el producto del costo del camino que el flujo atraviesa $r(\rho_T^{ij})$ por el costo de la correspondiente capacidad seleccionada para el camino $T(b_{ij})$ por kilómetro.

  El resto de las ecuaciones son restricciones.

- La segunda ecuación es la primer restricción. Indica que cada link entre nodos edges $e \in E_D$ que forma parte de la solución, es decir, que $b_{ij} \neq 0$, debe tener un único flujo asociado en G_T.

- La segunda restricción indica que cada link entre un nodo de acceso $v_i \in V_A$ y un nodo edge $v_j \in V_E$ que es parte de la solución ($b_{ij} \neq 0$) debe ser implementado con dos flujos, si los nodos están en diferentes estaciones de red ($ns(v_i, v_j) = 0$).

- La tercer restricción establece que cada nodo de acceso $v_i \in V_A$ es conectado a un único nodo edge en la solución. Solo los nodos edges que son vecinos $v_j \in V_E$ con $b_{ij} \neq 0$ son considerados y la suma debe ser uno.

- La cuarta restricción expresa que ningún nodo de acceso $v_i \in V_A$ es conectado a otro nodo de acceso en la solución.

- La quinta restricción significa que la conexión desde un nodo de acceso $v_p \in V_A$ a su correspondiente nodo edge $v_q \in V_E, b_{pq} \neq 0$, debe tener capacidad suficiente para satisfacer la suma de demandas de tráfico, comprometido y eventual, que parten de v_q hacia otros nodos en la red.

- La sexta y séptima restricción definen las matrices de demanda $\{\bar{M}'_0, ..., \bar{M}'_{|E_T|}\}$. La sexta lo establece para el escenario nominal ($t = 0$) y la séptima para el resto de los escenarios de falla ($t \in E_T$). En cada caso se asocian como demanda efectiva entre $v_i, v_j \in V_E$: la propia ($\vec{m}_{ij}$) más la de los nodos de acceso que por estar conectados a v_i ($v_e \in V_A$ con $b_{ei} \neq 0$) o a v_j ($v_f \in V_A$ con $b_{fj} \neq 0$), deben cursar el tráfico entre ellos o hacia estos edges a través de túneles que los últimos comparten.

  Para los casos de falla ($t \in E_T$), se obvian aquellos nodos de acceso cuyos caminos de transporte intersecten a la arista en falta ($\Psi(ei) \cap \{t\}$ y $\Psi(fj) \cap \{t\}$).

- La octava expresa que en cualquier escenario de falla ($t \in E_T$), el ruteo propuesto ($\rho_{D_t} = \Phi(t)$), no debe hacer uso de la arista que ha fallado ($|\rho_{D_t} \cap t| = 0$).

- La novena indica que en cada escenario de falla y en el nominal ($t \in \{\emptyset \cup E_T\}$), el escenario de ruteo correspondiente ($\rho_{D_t} = \Phi(t)$), debe incluir un único túnel entre cualesquiera dos nodos edges: $v_i, v_j \in V_E$, que tengan demanda efectiva entre si para ese escenario en particular ($\vec{m}'^t_{ij} \neq \vec{0}$).

- La décima y última restricción cumple una función similar a la quinta, pero entre los nodos edges. Es algo más complicada pues en una arista entre edges ($e \in E_E$) no solo pasa el tráfico entre estos sino el tráfico en tránsito entre otros para ese escenario $t \in (\emptyset \cup E_T)$.

  La idea es que cuando cualquier tunel para este escenario $\rho^{ij}_{D_t}$ toca la arista e (equivalente a $|\rho^{ij}_{D_t} \cap e| = 1$), se cargan a e las demandas correspondientes.

2.. Modelo de Conexión Acceso-Edge

En esta sección se detalla la hipótesis hecha, el modelo creado y la solución propuesta para conectar los nodos de acceso V_A a los nodos edges V_E.

Un nodo de acceso es conectado a uno y solo un nodo de datos. Si el nodo al cual se conecta es también un nodo de acceso no tendría sentido pues sería imposible que ambos nodos mandaran tráfico a otros nodos aparte de ellos ya que no se permite más de una conexión para cada uno de

ellos y por lo tanto quedarían aislados del sistema. Por tanto, un nodo de acceso siempre se encuentra conectado un nodo edge.

Si el nodo de acceso y el nodo edge están en la misma estación de red entonces la conexión entre estos se hace localmente. De lo contrario harán uso de la Red de Transporte para conectarse entre ellos.

Un punto escencial a determinar es si las conexiones acceso-edge deberían ser protegidas a fallas en el transporte o no. Si el modelo a aplicar es el primer caso entonces será necesario tener dos caminos independientes en transporte, es decir, dos caminos arista-disjuntos, entre el nodo acceso y el correspondiente nodo edge.

Dado que la topología sobre la cual se trabajó es básicamente una topología anillada donde cada anillo es contiguo a otro anillo, si el objetivo es llegar a un nodo edge cercano por dos caminos independientes en transporte será necesario recorrer ambos arcos del anillo. Esto presenta algunas desventajas. En primer lugar, el costo es superior comparado con la opción de que ambos flujos utilicen el camino más corto. Esto es aún peor si se tiene en cuenta que el camino alternativo puede ser 100 veces más largo que el camino más corto. En segundo lugar, la capacidad del anillo que contiene a ambos nodos se consume completamente.

Por otro lado, imaginemos un modelo con conexiones de acceso-edges desprotegidas donde un nodo edge es elegido de acuerdo al criterio del vecino más cercano. En la Tabla 4.1 y en la Figura 4.1 puede verse un ejemplo de 5 nodos A, B, C, D y E. Los nodos de acceso B y C son asociados con el nodo edge A porque es el más cercano en distancia que el nodo E. Sin embargo, el nodo D se asocia con el nodo E. Notar que los nodos A y E son importantes y tienen mucho tráfico a enviar. Se puede ver que el link de transporte correspondiente a $A - B$ concentra el tráfico de B y C mientras que el link de transporte $C - D$ no tiene carga. Se puede concluir de este ejemplo que siguiendo este procedimiento, la carga en la Red de Transporte resultaría desbalanceada y un error en la precisión del cálculo del costo de la red aparecería pues según la definición 4.15, se asume una utilización balanceada de la capacidad, que significa que cada link de transporte de un camino llevan la misma carga de flujo. Además, es importante que el camino entre dos estaciones importantes lleve la misma capacidad. Consecuentemente, la capacidad del path $A - E$ debería consistir de 4 partes de 38 VC12, sumando un total de 152 VC12, pero la carga real del camino sería en realidad 38 + 8 + 20 = 66 VC12, lo cual es menor que la mitad de la carga. El ejemplo se torna aún peor cuanto más nodos de acceso existan en el medio.

Station	$\hat{\mathbf{b}} \in \hat{\mathbf{B}}$
A	322
B	30
C	8
D	20
E	184

Tabla 4.1: Estaciones y capacidades

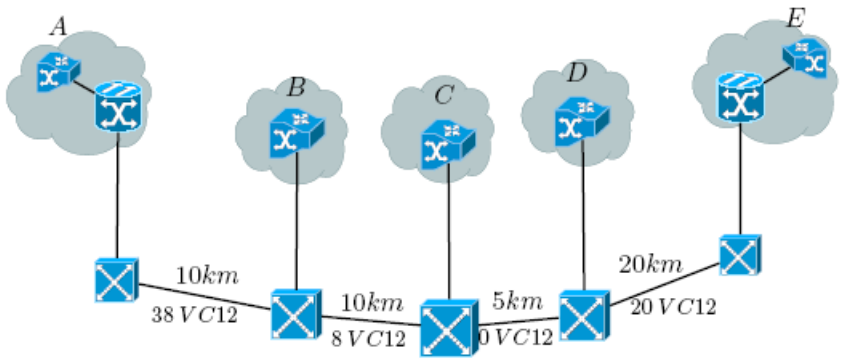

Figura 4.1: Red de Transporte desbalanceada

Luego de introducir las ideas anteriores resumiremos las suposiciones que se tomaron para la solución al problema de conectar nodos de acceso a nodos edges y sus beneficios. El primer paso consiste en encontrar un camino para asignar las demandas de los nodos de acceso a los nodos edges de modo de evitar problemas como el mencionado anteriormente. Esto significa que en los sucesivo, una vez llevado a cabo este proceso, estaremos trabajando con una red reducida donde los nodos de acceso habrán sido eliminados y sus demandas asociadas serán transferidas a los nodos edges con los cuales se han conectado.

Con el objetivo de hacer frente al problema de las capacidades desbalanceadas se hará una partición de cada nodo en una estación en dos nodos acceso con la mitad de la demanda real. Cada uno de ellos serán conectados al nodo edge más cercano. En la Tabla 4.2 y la Figura 4.2 se continúa con el ejemplo previo. Los nodos B, C y D envían la mitad de sus demandas a A y E. dejando la misma utilización de tráfico en cada arista de transporte. La capacidad de cada arista entonces deberá ser diseñada con 4 partes de 29 VC12. Con esta perturbación, la carga en la Red de Transporte no varía demasiado y también se espera que la solución para G_E no cambie, o en su defecto, que el impacto sea mínimo.

Station	$\hat{\mathbf{b}} \in \hat{\mathbf{B}}$	**Station**	$\hat{\mathbf{b}} \in \hat{\mathbf{B}}$
B_1	15	B_2	15
C_1	4	C_2	4
D_1	8	D_2	8

Tabla 4.2: Estaciones y capacidades balanceadas.

Tomando en cuenta el análisis anterior, el modelo sobre el cual trabajaremos tiene las conexiones acceso-edge resuelta. En otras palabras, estamos en condiciones de omitir los nodos de acceso obteniendo así una simplificación del problema. La red resultante tiene solo nodos edges con el tráfico asociado a ellos que se corresponde con los tráficos provenientes de los nodos de acceso conectados a cada uno de estos. Estas suposiciones reducen sustancialmente la dificultad para la construcción de algoritmos sin pérdida de generalidad en el modelo, precisión en los resultados o representación

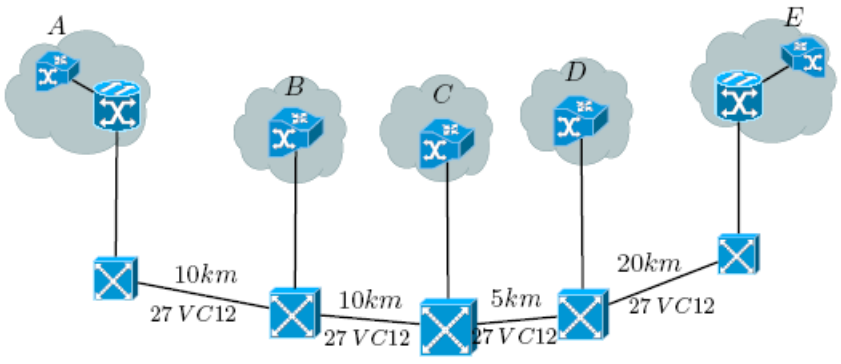

Figura 4.2: Red de Transporte balanceada

de la realidad.

De ahora en más, se tendrán en cuenta todas estas suposiciones y pasaremos a referirnos al problema como *MORN-Reducido* (RMORN).

2..1. Modelo matemático MORN-Reducido

El modelo simplificado elimina unas cuantas ecuaciones obteniendo entonces la siguiente formulación:

$$
RMORN \begin{cases}
\min\limits_{B,\Phi,\Psi} \sum\limits_{\substack{\rho_T=\Psi(e) \\ e=(v_i,v_j) \\ e\in E_D}} r(\rho_T^{ij})T(b_{ij}) & \\
|\rho_T^{ij}| = 1 & \forall e\in E_E, e=(v_i,v_j), \\ & b_{ij}\neq 0, \rho_T=\Psi(e). \\
|\rho_{D_t} \cap t| = 0 & \forall t\in E_T, \rho_{D_t}=\Phi(t). \\
|\rho_{D_t}^{ij}| = 1 & \forall v_i,v_j\in V_E, \\ & \forall t\in\{\emptyset\cup E_T\}, \\ & \rho_{D_t}=\Phi(t), \vec{m}_{ij}\neq\vec{0}. \\
b_{pq} \geq \sum\limits_{\forall v_i,v_j\in V_E} \left(\dot{m}_{ij}|\rho_{D_t}^{ij}\cap e|\right) + z_Q \left(\sum\limits_{\forall v_i,v_j\in V_E} \left(\ddot{m}_{ij}|\rho_{D_t}^{ij}\cap e|\right)\right) & \forall e\in E_E, e=(v_p,v_q), \\ & \forall t\in\{\emptyset\cup E_T\}, \\ & \rho_{D_t}=\Phi(t), \\ & \vec{m}_{ij}=(\dot{m}_{ij},\ddot{m}_{ij}).
\end{cases}
$$

Se puede observar que este modelo matemático es mucho más simple que el anterior. Consiste en encontrar una red de costo mínimo con:

- Un camino en G_T para enrutar cada arista en G_E.
- Esquemas de ruteo para cada escenario simple de falla sin utilizar la arista de transporte que falló.
- Un túnel en G_E entre dos nodos de datos $v_i, v_j \in V_E$ que intercambian tráfico en escenarios nominal o de fallas de G_T.

- Una selección de capacidades para los links en E_E donde los requerimientos de demanda para la configuración de túneles escogida es garantizada.

En el siguiente capítulo se estudiará la complejidad computacional del problema MORN. Para ello se hará una introducción a los principales conceptos y posteriormente nos abocaremos a demostrar que el problema MORN pertenece a la clase de problemas **NP-Hard**.

Capítulo 5

Complejidad Computacional

El objetivo de este capítulo es ofrecer al lector los fundamentos necesarios para comprender los conceptos de complejidad computacional. Es sabido que para muchos problemas computacionales es posible encontrar un algoritmo eficiente que lo resuelva, mientras que existen otro tipo de problemas complejos para los cuales no se ha podido encontrar una solución en tiempos de computos prudentes. MORN pertenece a esta clase de problemas. Se demostrará formalmente que el MORN es un problema NP-Hard [5].

1.. Introducción

La complejidad de un algoritmo es medido en términos de complejidad de *tiempo* y *espacio*. Ambos están relacionados con la cantidad de entradas, usualmente conocido como el tamaño del problema. La complejidad de tiempo está asociado al número de operaciones aritméticas elementales necesarias para ejecutar el algoritmo.
La complejidad del espacio es la cantidad de celdas de memoria que un algoritmo necesita durante su ejecución.

La dificultad de un problema está relacionada con su estructura y el tamaño de la instancia a ser considerada. En teoría de grafos, este tamaño se expresa generalmente como el número de vértices del grafo y las aristas.

La teoría de la complejidad computacional es aplicada a *Problemas de Decisión*. Un problema de decisión es un tipo especial de problema de computación cuya respuesta es, o bien *SI* o bien *NO*. Un problema de decisión π consiste de un conjunto de instancias D_π y un subconjunto de instancias $Y_\pi \subset D_\pi$ para las cuales su respuesta es *SI*.

Los problemas de optimización combinatoria como el que se presenta en este trabajo presentan un espacio de soluciones factibles enumerable. Dado que es posible relacionar este tipo de problemas con problemas de decisión, la teoría es aplicable a nuestro caso.

2.. P y NP

Definición 5.1. *Un problema $\pi \in$ **P** si existe un algoritmo de complejidad polinomial para resolverlo, o en otras palabras, el número de pasos para resolverlo es menor que una función polinomial dependiente de n siendo n el tamaño del input.*

Definición 5.2. *Un problema $\pi \in$ **NP** si es posible verificar en tiempo polinomial que una instancia cuya respuesta es* SI *es efectivamente* SI.

Si bien para los problemas en **P** existe un algoritmo eficiente, no sabemos que tan complicado es encontrar una solución para un problema en **NP**. Claramente, $P \subset NP$ pues si es viable resolver un problema en tiempo polinomial, una solución en este conjunto debe ser verificada también en tiempo polinomial. Por otro lado, existe una conjetura vigente a la fecha[1] sobre si **P** $=$ **NP**. La teoría explicada en la siguiente sección está basada en la suposición de que la ecuación anterior no se cumple.

3.. NP-Completitud

La idea clave en esta teoría es el concepto de transformación polinomial:

Definición 5.3. *Una reducción polinomial desde un problema de decisión π' a π es una función $f : D_{\pi'} \to D_\pi$ tal que puede ser computada en tiempo polinomial y para cada $d \in D_\pi, d \in Y_{\pi'} \iff f(d) \in Y_\pi$. Denotamos $\pi' \preccurlyeq \pi$ si existe una transformación polinomial de π' a π.*

Puede verse la importancia de la reducción polinomial en el siguiente lema:

Lema 5.1. *Si $\pi' \preccurlyeq \pi$ entonces si $\pi \in$ **P** implica que $\pi' \in$ **P**.*

De este lema deducimos que se puede determinar la complejidad de un problema dado si es posible encontrar una reducción polinomial de este problema a otro problema del cual se conoce su complejidad.

Los problemas **NP-Completos** son considerados los problemas más difíciles de la clase **NP** y posiblemente no formen parte de la clase **P**. Son aquellos problemas que forman parte de un subconjunto de los **NP** tal que todo problema en **NP** puede ser reducido por una transformación polinomial a ellos.
Matematicamente, un problema π es **NP-Completo** si

- $\pi \in$ **NP**.
- Para cada $\pi' \in$ **NP**, $\pi' \preccurlyeq \pi$.

Un problema que satisface la segunda condición pertenece a la clase NP-Hard independientemente de que se satisfaga la primera. Esto significa que dicho problema es al menos tan difícil como todos los problemas en **NP**.

[1] Es uno de los siete problemas matemáticos abiertos anuciados por el Clay Mathematic Institute recompensados con las suma de un millón de dólares.

4.. Complejidad del Algoritmo

En esta sección se demostrará que el problema **RMORN** es un problema NP-Hard.

4..1. Condiciones Para la 2-Arista-Conectividad

Teorema 4..1.1. *Sean la redes $G_T = (V_T, E_T)$ y $G_D = (V_D, E_D)$ en las condiciones del* MORN-Reducido*. Entonces una topología 2-arista-conexa en ambas redes es una condición necesaria para la existencia de una solución factible*

Prueba.

(A) $G_T = (V_T, E_T)$ debe ser 2-arista-conexo.

Primero, supongamos que G_T es no conexo. Entonces existen al menos dos componentes conexas disjuntas H_1 y H_2 incluídas en G_T.
Sea $\hat{i}, \hat{j} \in V_D$ tal que $tns(\hat{i}) \in H_1$ y $tns(\hat{j}) \in H_2$. Si $m_{\hat{i}\hat{j}} \neq 0$ no hay manera de enrutar tráfico desde $\hat{i}$ hacia $\hat{j}$. Si $m_{\hat{i}\hat{j}} = 0 \ \forall \hat{i}, \hat{j} \in V_D$ tal que $tns(\hat{i}) \in H_1$ y $tns(\hat{j}) \in H_2$, entonces el problema podría separarse en subproblemas equivalentes.

Ahora supongamos que G_T es un grafo conexo pero no 2-arista-conexo. Entonces existe un link $\hat{e} \in E_T$ que es un puente en G_T.
Sea H_1 y H_2 las componentes conexas disjuntas resultantes de quitar el link $\hat{e}$ de G_T, entonces $G_T = H_1 \biguplus H_2$.
Sea $\hat{i}$, $\hat{j} \in V_D$ tal que $m_{\hat{i}\hat{j}} \neq 0$ y tal que $tns(\hat{i}) \in H_1$, $tns(\hat{j}) \in H_2$. (De otra forma, el problema podría separarse en dos subproblemas equivalentes.)

Sea $\rho_{D_{\hat{e}}}^{\hat{i}\hat{j}}$ el tunel de $\hat{i}$ a $\hat{j}$ en G_D en el escenario de falla $\hat{e}$. Es claro que $\exists e = (u, v) \in \rho_{D_{\hat{e}}}^{\hat{i}\hat{j}}$ en el cual $\hat{e} \in \rho_T^{tns(u)tns(v)}$.
Si este no fuese el caso, considerando:

$$\hat{p} \ = \bigcup_{\forall (p,q) \in \rho_{D_{\hat{e}}}^{\hat{i}\hat{j}}} \{(x, y) : (x, y) \in \rho_T^{tns(u)tns(v)}\} \ .$$

podría ser un camino que conectara $tns(\hat{i})$ con $tns(\hat{j})$ en $G_T \backslash \{\hat{e}\}$.
Esto contradice el hecho de que $\hat{e}$ es una arista puente en G_T. Por lo tanto, G_T es 2-arista-conexo.

(B) $G_D = (V_D, E_D)$ debe ser 2-arista-conexo.
Recordar que hemos asumido $V_D = V_E$ (los nodos edges integran como propios los flujos de los nodos de acceso conectados a cada uno de ellos).
Supongamos nuevamente que G_D es conexo (de otro modo el problema podría particionarse en subproblemas equivalentes), pero no 2-arista-conexo. Entonces existe un link $\hat{e}_D \in E_D$ tal que es una arista puente en G_D.

Sea Y_1 e Y_2 las componentes conexas disjuntas resultantes de quitar dicha arista $\hat{e}_D$ en G_D. Entonces se tiene que $G_D = Y_1 \biguplus Y_2$.
Sea $\hat{i}, \hat{j} \in V_D$ tal que $m_{\hat{i}\hat{j}} \neq 0$ y tal que $\hat{i} \in Y_1, \hat{j} \in Y_2$.
Sea $\rho_{D_0}^{\hat{i}\hat{j}}$ el tunel de $\hat{i}$ a $\hat{j}$ en G_D en el escenario nominal.
Denotemos $\hat{e}_D = (u, v)$. Sea $\rho_T^{tns(u)tns(v)}$ el ruteo de la arista $\hat{e}$ en G_T y elijamos cualquier $\hat{e}_T \in \rho_T^{tns(u)tns(v)}$.
Entonces en el escenario de falla $\hat{e}_T$, el grafo G_D es separado en las componentes Y_1 e Y_2. Entonces el tunel $\rho_{D_{\hat{e}_T}}^{\hat{i}\hat{j}}$ entre $\hat{i}$ y $\hat{j}$ no podría existir en el escenario de falla $\hat{e}_T$. Por tanto, G_D no puede tener aristas puentes y es 2-arista-conexo.

□

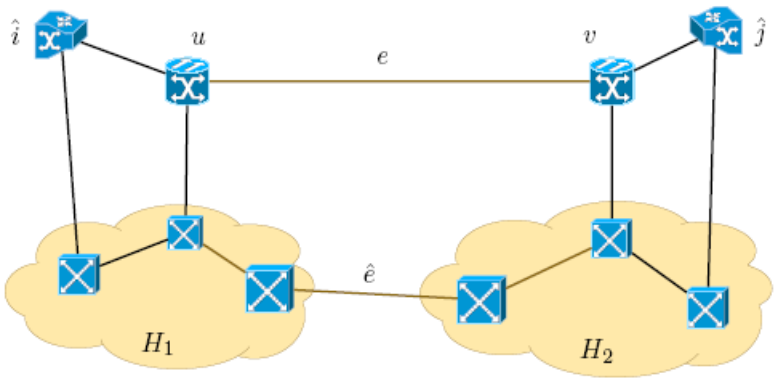

Figura 5.1: Elementos referenciados en la prueba del Teorema.

4..2. Problemas de Complejidad Similar

A continuación reduciremos instancias particulares del **RMORN** a problemas conocidos y que pertenecen a la clase NP-Hard. Estos problemas son el **2-ECSS** (2-Edge-Connectivity Problem) y el **MBA** (Minimum Biconnectivity Augmentation), ambos definidos más adelante. Finalmente, mostraremos que una versión reducida del **RMORN** es equivalente al problema **MCCST** (Minimum Communication Cost Spanning Tree).

Definición 5.4. *2-ECSS (2-Edge-Connectivity Problem)*
*Dado un grafo arbitrario no dirigido $G = (V, E)$, el 2-Edge-Connectivity Problem (denotado **2-ECSS**) consiste en encontrar un conjunto de cardinalidad mínimo $U \subseteq E$ tal que el subgrafo $\hat{G} = (V, U)$ es 2-arista-conexo.*
Este problema es NP-Hard [6] [19] [10].

Teorema 4..2.1. *Existe una instanciación del **RMORN** equivalente a una instancia genérica del problema **2-ECSS**.*

Prueba.
Consideremos el problema **RMORN** con la siguiente parametrización:

1. $\bar{M}_0 = \bar{M}_e \ \forall e \in E_T$ (los tráficos en los escenarios de falla y el escenario nominal son los mismos).

2. $\bar{M}_0 = \{m_{ij}\}_{i,j \in V_D}$ es tal que $\dot{m}_{ij} = \ddot{m}_{ij} = 1 \ \forall i, j \in V_D$.

3. $\hat{B} = \{b_0, b_1\}$ con $b_0 = 0$ y $b_1 \gg \sum_{i,j \in V_D} 2\dot{m}_{ij}$, o sea $b_1 \gg \frac{2|V_D| \times (|V_D|-1)}{2} = |V_D| \times (|V_D| - 1)$.

4. G_T y G_D son tal que G_D "copia exáctamente" la red de transporte G_T. Se asume G_T como 2 arista-conexa.

5. Definiremos $T(\cdot)$ como sigue:

$$T(b_{ij}) = \begin{cases} 1/r(\rho_T^{ij}) & si\ b_{ij} = b_1 \\ \\ 0 & sino \end{cases}$$

   De esta forma en la función objetivo del **RMORN** cada término sumando $r(\rho_T^{ij}) \times T(b_{ij})$ cumplirá:

$$r(\rho_T^{ij}) \times T(b_{ij}) = \begin{cases} 1 & si\ b_{ij} \neq 0 \\ \\ 0 & sino \end{cases}$$

6. $c = \{r(e)\}_{e \in E_T}$ es tal que $r(e) = 1 \ \forall e \in E_T$.

En estas condiciones el **RMORN**, con esta instanciación en sus parámetros y definición de $T(\cdot)$, es equivalente al problema **2-ECSS**.

La solución óptima de esta instanciación del **RMORN** es una terna $\tilde{S} = (\tilde{M}_D, \tilde{M}_T, \tilde{B})$ donde:

- $\tilde{M}_D$ es la copia exacta de $\tilde{M}_T$ en G_D.
- $\tilde{B}$ = $\{b_{ij} = b_1 : (i,j) \in \tilde{M}_D\}$.

Las condiciones 3, 4 y 5 aseguran la resistencia a fallas simples en la red de transporte por parte de $\tilde{S}$ y además el costo en la función objetivo del **RMORN** es solamente influenciado por la elección de las aristas de datos puesto que por condición 5, el costo correspondiente a cada arista $\hat{e}_d \in E_D$ elegida será de 1.
(Si $\hat{e}_d = (\hat{i}, \hat{j}) \in E_D$ es elegido, entonces $T(b_{\hat{i}\hat{j}}) \times r(\rho_T^{\hat{i}\hat{j}}) \overset{df\ T(\cdot)}{=} (1/r(\rho_T^{\hat{i}\hat{j}})) \times r(\rho_T^{\hat{i}\hat{j}}) = 1$).

Esto implica que la solución óptima global $\tilde{S}$ tendrá en $\tilde{M}_D$ la mínima cantidad de aristas de E_D de forma que $\tilde{M}_D$ cubra a V_D en forma 2-arista-conexa, con lo cual $\tilde{M}_D$ es óptimo global de **2-ECSS**(V_D, E_D).

Dado que **2-ECSS** es un problema NP-Hard entonces esta instanciación del **RMORN** es un problema NP-Hard [6] [19] [10]. □

Definición 5.5. *MBA (Minimum Biconnectivity Augmentation)*
Sea un grafo $G = (V, E)$ y una función que llamaremos "peso" $w : V \times V \to \mathbb{N}$. El objetivo de este problema es encontrar un conjunto E' para G, es decir, un conjunto E' de pares de vértices no ordenados de V tal que $G = (V, E \cup E')$ es biconexo y $\sum_{(u,v)\in E'} w(u, v)$ es mínimo [7] [8].

Teorema 4..2.2. *Existe una instanciación del **RMORN** equivalente al problema **MBA** en su forma genérica.*

Prueba.
Consideremos una instancia del **RMORN** satisfaciendo los siguientes puntos :

1. G_D "copia" a G_T topológicamente (Asumiendo G_D y G_T completos).

2. Sea $H_D = (V_D, \hat{U}_D) \subset G_D$ un subgrafo fijo tal que H_D **no** es 2-arista-conexo.

3. Sea $H_T = (V_T, \hat{U}_T)$ la "copia" topológica de H_D en G_T, es decir:
   $V_T = \{tns(v) : v \in V_D\}$, $\hat{U}_T = \{(tns(u), tns(v)) : (u, v) \in \hat{U}_D\}$.

4. $C = \{c_e\}_{e\in E_T}$ tal que
$$c_{ij} = \begin{cases} w_{ij} & si\ (i,j) \notin \hat{U}_T \\ 0 & sino \end{cases}$$
   $W = \{w_{ij}\}_{(i,j)\in\hat{U}_T}$, con W satisfaciendo desigualdad triangular y C satisfaciendo desigualdad triangular.

5. $\bar{M}_0 = \bar{M}_e \quad \forall e \in E_T$ tal que $\dot{m}_{ij} = 1 \quad \forall i, j \in V_D$, $\ddot{m}_{ij} = 0 \quad \forall i, j \in V_D$.

6. $\hat{B} = \{b_0, b_1\}$ tal que $b_1 \gg \sum_{i,j\in V_D} 2\dot{m}_{ij} = |V_D| \times (|V_D| - 1)$.

7.
$$T(b_{ij}) = \begin{cases} 0 & si \quad (i,j) \in \hat{U}_D \\ 1 & si \quad b_{ij} = b_1,\ (i,j) \notin \hat{U}_D \\ 0 & si \quad b_{ij} = b_0\ (i,j) \notin \hat{U}_D \end{cases}$$

En estas condiciones el problema **RMORN** es equivalente al problema **MBA**$(V_T, \hat{U}_T, C)$, es decir, encontrar $Y_T \subset E_T \backslash \hat{U}_T$ tal que $\hat{G}_T = (V_T, \hat{U}_T \cup Y_T)$ es 2-arista-conexo a costo mínimo.

Sea $\tilde{S} = (\tilde{H}_T, \tilde{H}_D, \tilde{B})$ solución óptima del **RMORN** con esta parametrización. Se cumple que:

- $\tilde{H}_D$ es la "copia" topológica de $\tilde{H}_T$ en G_D.
- $\hat{B} = \{b_{ij} = b_1 : (i,j) \in \tilde{H}_D\}$.

De las condiciones 5 y 7, hay solo una capacidad tecnológica disponible para los links de datos utilizados y sus costos son iguales a 1. Esto implica que al dimensionar $\tilde{H}_D$ con $\tilde{B}$, el costo de la solución **no** es influenciado por la variable tecnológica. Esto hace que el costo de la función objetivo del **RMORN** sea solamente afectado por las aristas de transporte de $\tilde{H}_T$ que subyacen a los de $\tilde{H}_D$.

La condición 6 garantiza que ante la caída de $\hat{e}_T \in \tilde{H}_T$ en $\hat{H}_D = \tilde{H}_D \backslash \{\hat{e}_D\}$ ($\hat{e}_D$ es el link de datos afectado por $\hat{e}_T$) habrá un tunel capacitado para cada demanda $\dot{m}_{\hat{i},\hat{j}}$, $\hat{i}, \hat{j} \in V_D$.
Claramente, $\tilde{S}$ satisface la resistencia a fallas simples en la red de transporte.

Ahora bien, analicemos la función objetivo en $\tilde{S}$.

$$\sum_{(i,j)\in\tilde{H}_D} r(\rho_T^{ij}) \times T(b_{ij}) \quad = \quad \sum_{\substack{(i,j)\in\tilde{H}_D \\ (i,j)\in E_D\backslash\tilde{U}_D}} r(\rho_T^{ij}) \quad =$$

$$\sum_{\substack{(i,j)\in\tilde{M}_D \\ (i,j)\in E_D\backslash\tilde{U}_D}} r(tns(i), tns(j)) \quad = \quad \sum_{\substack{e_T\in\tilde{H}_T \\ e_T\in E_T\backslash\tilde{U}_T}} r(e_T)$$

.

$\tilde{H}_T$ tiene una estructura de la forma $\tilde{H}_T = (V_T, \hat{U}_T \cup \hat{E}_T)$ con $\hat{E}_T \subset E_T \backslash \hat{U}_T$, donde $\hat{E}_T$ es el conjunto de aristas de transporte que hacen que $\tilde{H}_T$ sea 2-arista-conexo de costo mínimo estando fijas las aristas de $\hat{U}_T$; ellas minimizan el costo de envío de tráfico directo $\dot{m}_{\hat{i},\hat{j}}$, si $(\hat{i}, \hat{j}) \in \hat{U}_D$. Luego $\tilde{H}_T$ es solución optima global de **MBA**$(V_T, \hat{U}_T, C)$.

Dado que el **MBA** es un problema NP-Hard, esta instanciación del **RMORN** es un problema NP-Hard. □

Definición 5.6. *MCCST (Minimum Communication Cost Spanning Tree)*
Sea $G = (V, E, w)$ un grafo no dirigido con w una función no negativa que determina la distancia de las aristas. Se conocen también una función requerimiento denotada como $\lambda(u, v)$ para cada par de vértices. El costo de comunicación entre dos vértices es el requerimiento multiplicado por la distancia del camino entre estos vértices. El objetivo de este problema es construir un spanning tree *con mínimo costo de comunicación, o sea, T tal que $\sum_{u,v\in V} \lambda(u, v) d_T(u, v)$ es mínimo, siendo $d_T(u, v)$ la suma de los w que componen el camino de u a v.*

Nota 5.1. *Relajando en **RMORN** la restricción de resistencia a fallas simples en la red de transporte, el problema resultante, al cual llamaremos **RRMORN**, es NP-Hard.*

Prueba.
Consideremos una instancia del **RMORN** con la siguiente parametrización:

1. G_D "copia" a G_T topológicamente.
2. $\bar{M}_0 = \bar{M}_e \ \ \forall e \in E_T$, tal que $\dot{m}_{ij} \in N \ \ \forall i,j \in V_D$
$$\ddot{m}_{ij} = 0 \ \ \forall i,j \in V_D.$$
3. $\hat{B} = \{0,1,2,...\} = \mathbb{N}$
Las capacidades tecnológicas son enteros positivos y en principio pueden tener cualquier valor en $\mathbb{N}$.
4. La función $T(\cdot)$ fija de antemano el dimensionamiento y vale:
$$\begin{cases} T(b_{ij}) = b_{ij} = {}^{2}\dot{m}_{ij} & \forall i,j \in V_D. \\ T(\emptyset) = 0 \end{cases}$$
.
5. Los costos $C = \{r_e\}_{e \in E_T}$ son reales positivos satisfaciendo la desigualdad triangular.

Al eliminar en el **RMORN** la restricción en la red de datos de resistencia a fallas simples en arcos de la red de transporte, ahora las soluciones topológicas factibles minimales del **RRMORN** serán árboles que cubren V_D dimensionados de antemano vía la matriz $\bar{M}_0$ ($T(b_{ij}) = \dot{m}_{ij}$).
Además la función objetivo ahora es de la forma:

$$\sum_{\forall i,j \in V_D} r(\rho_T^{ij}) \times T(b_{ij}) \ \ = \ \ \sum_{\forall i,j \in V_D} r(\rho_T^{ij}) \times \dot{m}_{ij}. \quad .$$

donde $r(\rho_T^{ij})$ es el costo del camino en Υ_T entre $tns(i)$ y $tns(j)$, con Υ_T el árbol "copia" topológica de Υ_D, siendo Υ_D el árbol de cubrimiento de V_D en G_D solución del **RRMORN**.
El **RRMORN** es equivalente entonces al problema Minimum Communication Cost Spanning Tree **(MCCST)** definido anteriormente.
Ahora bien, dado que el **MCCST** es un problema NP-Hard [20, 21] entonces concluímos que el **RRMORN** es también NP-Hard. □

[2]Pre-asignamos a b_{ij} el valor de la demanda comprometida entre $i,j \in V_D$

Parte III

Metaheurísticas

Capítulo 6

Introducción - Tabú Search

En este capítulo se explicará en detalle los aspectos teóricos estudiados para la resolución del problema mediante un enfoque metaheurístico.
La resolución al problema puede dividirse básicamente en dos partes; la primer parte en la cual nos abocamos a la búsqueda de una solución inicial factible que nos sirva como punto de partida para la resolución del problema, el cual denotamos como **RMORN** y la segunda parte que toma esta solución inicial obtenida y le aplica una optimización metaheurística basada en Tabú Search.
El propósito de este capítulo es brindar de forma sucinta una introducción al concepto de metaheurísticas y a una descripción en detalle del Tabú Search.
En los siguientes dos capítulos explicaremos en detalle la resolución al problema **RMORN** y la optimización basada en la metaheurística Tabú Search, respectivamente.

Comenzamos este capítulo entonces con una introducción teórica al mundo de las metaheurísticas y en especial, al Tabú Search.

1.. Metaheurísticas

Las metaheurísticas son procedimientos genéricos de exploración del espacio de soluciones para problemas de optimización y búsqueda. Son procedimientos iterativos que guían una heurística subordinada combinando en forma inteligente distintos conceptos para explorar y explotar adecuadamente el espacio de búsqueda. Las metaheurísticas brindan un mecanismo por el cual podemos encontrar buenas soluciones a un problema en tiempos de ejecuciones razonables aunque estas soluciones no sean óptimas. Algunas ventajas de las metaheurísticas son:

- Son algoritmos de propósito general.
- Son exitosos en la práctica.
- Por lo general son fácilmente implementables.
- Pueden paralelizarse facilmente.

Como desventajas encontramos:

- No son algoritmos exactos sino aproximados, no necesariamente llegan a una solución óptima.
- Son altamente no determinísticos.
- La mayoría presenta poca base teórica.

Para obtener buenas soluciones cualquier algoritmo debe lograr contemplar y balancear dos características importantes, *Intensificación* y *Diversificación*

Definición 6.1. *Intensificación se entiende como la cantidad de esfuerzo empleado en la búsqueda de la región actual (explotación del espacio).*

Definición 6.2. *Diversificación entendemos como la cantidad de esfuerzo empleado en la búsqueda de regiones distantes del espacio (exploración).*

Este equilibrio entre ambas características es importante para identificar rápidamente regiones del espacio de soluciones de buena calidad y además para no consumir mucho tiempo en regiones del espacio prometedoras o ya exploradas.
Las metaheurísticas aplican distintas estrategias para obtener un buen balance entre intensificación y diversificación.

1..1. Clasificación de las Metaheurísticas

Las metaheurísticas pueden clasificarse en diversos grupos de acuerdo a sus características. Una posible clasificación puede ser la siguiente:

- *Constructivas.* Estas heurísticas parten de una solución inicial vacía y van añadiendole componentes hasta construir una solución. Ejemplos de estas heurísticas son *GRASP*[29], *Ant Colonies*[30].
- *Basadas en trayectorias.* Parten de una solución inicial e iterativamente tratan de reemplazarla por otra solución de su vecindario de mejor calidad. Ejemplos de estas heurísticas son *Búsqueda Local*, *Simulated Annealing*[31], *Tabú Search*[32].
- *Basadas en poblaciones.* Evolucionan una población de soluciones iterativamente. La característica fundamental de este tipo de metaheurísticas es la interacción entre los miembros de la población en contraste con las búsquedas locales. Un ejemplo de estas metaheurísticas son los *Algoritmos Evolutivos*[33, 34].
- *Basadas en Memoria.* Estas metaheurísticas suelen almacenar cierta información que luego será utilizada en la toma de ciertas decisiones, en particular para determinar el próximo paso a seguir. Ejemplos de tales metaheurísticas son *Tabú Search*[32].

1..2. Tabú Search

Una de las metaheurísticas más conocidas y empleadas en problemas de optimización es conocida como Tabú Search [32]. Esta metaheurística pertenece a la categoría de metaheurísticas con memoria y permite superar algunos inconvenientes de la búsqueda local convencional.

Permite resolver una amplia gama de problemas prácticos en diversas ramas como telecomunicaciones, scheduling, análisis financiero, logística, análisis biomédico.

La palabra Tabu tiene su origen en una lengua hablada por aborígenes de la Polynesia, donde era utilizada para indicar cosas que no podían tocarse porque eran consideradas sagradas. La palabra tabu también significa una prohibición impuesta por costumbres sociales como medidas de protección. La analogía en los problemas de optimización y en Tabú Search (TS) concretamente es que existirán elementos que serán considerados tabu durante la ejecución del algoritmo y que deberán evitarse. Existirá una analogía a la "memoria social"donde residirán estos elementos tabu. Al igual que en el contexto social, las prohibiciones pueden ser removidas de acuerdo a la ocasión o bien con el transcurso del tiempo. De modo que los elementos “tabú” en la búsqueda tabu también iran cambiando o se irán eliminando de acuerdo a cierto criterio durante el transcurso del tiempo.

1..3. Uso de Memoria

La estructura de memoria opera en cuatro dimensiones, recency, frequency, quality e influence. Las memorias basadas en recency y frequency se complementan mutuamente como veremos más adelante. La dimensión quality refiere a la capacidad para diferenciar soluciones de diferente calidad durante la búsqueda. En este contexto, la memoria puede utilizarse para identificar elementos y atributos que son comunes a soluciones de buena calidad y por consiguiente explotar los mismos para ir, gradualmente, mejorando las soluciones paso a paso, o bien identificar caminos para llegar a estas soluciones de buena calidad. O contrariamente, encontar elementos que contribuyen a malas soluciones y penalizarlos de modo de evitar tomar caminos que nos lleven a estas soluciones de poca calidad.
La cuarta dimensión, influence, considera el impacto de las elecciones hechas durante la búsqueda, no solo a nivel de calidad sino también de estructura. La flexibilidad de estas estructuras de memoria permiten guiar la búsqueda en un entorno multi-objetivo, donde la bondad de una dirección particular de búsqueda estará determinada por más de una función.

La memoria utilizada en TS es tanto explícita como de atributo. La primera registra soluciones completas que fueron visitadas durante la búsqueda. Una extensión de esta memoria almacena vecindades de soluciones elite, altamente atractivas pero que no han sido exploradas aún. Las soluciones memorizadas son utilizadas para expandir la búsqueda local. Alternativamente, TS utiliza memoria de atributos. Este tipo de memoria almacena información sobre atributos de soluciones que cambian cuando nos movemos de una solución a otra. A modo de ejemplo, en el diseño de una red, los atributos pueden ser nodos que se agregan/remueven u ordenan durante la búsqueda.

1..4. Intensificación y Diversificación

Dos componentes importantes que tiene en cuenta TS a la hora de llevar a cabo la búsqueda son la Intensificación y la Diversificación. La intensificación se basa en la modificación de los criterios de selección de modo de fomentar la combinación de movimientos o rasgos de soluciones que históricamente contribuyeron a buenas soluciones. Dado que las soluciones elite deben ser almacenadas para poder examinar sus vecindades, la memoria explícita está estrechamente relacionada con la implementación de estrategias de intensificación. La intensificación entonces consiste en examinar vecindades de soluciones elite. En este caso, el concepto de vecindad tiene un significado más amplio que el usual; esto es, además de considerar soluciones adyacentes o próximas a soluciones elite, las estrategias de intensificación generan vecinos mezclando componentes de distintas soluciones de buena calidad.

La etapa de diversificación por otro lado alienta al proceso de búsqueda a examinar regiones del espacio aún no visitadas y generar soluciones que difieran considerablemente de las vistas hasta el momento.

1..5. Fundamentos de TS y memoria de corto plazo

TS puede ser aplicado a muchos problemas de decisión sin la necesidad de transformar los mismos en su representación matemática. Sin embargo, es útil introducir una notación matemática para estos problemas como base para describir ciertas características de TS. Caracterizamos estos problemas como aquellos que buscan optimizar (maximizar, minimizar) una función $f(x)$ con $x \in X$ donde la función $f(x)$ puede ser lineal o no lineal y donde X resume ciertas restricciones en el vector de variables de decisión x.

Definición 6.3. *Una vecindad N es un conjunto de todas aquellas soluciones que pueden ser alcanzables a partir de una solución inicial por medio de un movimiento que puede ser un intercambio entre elementos que conforman la solución inicial.*

TS comienza del mismo modo que la búsqueda local ordinaria, procediendo iterativamente desde un punto (solución) hacia otro hasta que cierto criterio de terminación es alcanzado. Cada $x \in X$ tiene una vecindad asociada $N(x) \subset X$ y cada solución $x' \in N(x)$ es alcanzada desde x por una operación llamada *movimiento*.
La búsqueda local ordinaria o método de descenso simple busca minimizar la función $f(x)$ y en cada iteración permite movimientos dentro de la vecindad siempre y cuando la solución vecina x' tiene un costo menor que la solución actual x y finaliza cuando no se puede mejorar la solución. Esto tiene como deficiencia que el óptimo local encontrado no siempre será un óptimo global a la solución, o sea, no minimizará la función $f(x)$ sobre todo el espacio X.
En la siguiente Figura podemos ver gráficamente esta deficiencia de la búsqueda local ordinaria.

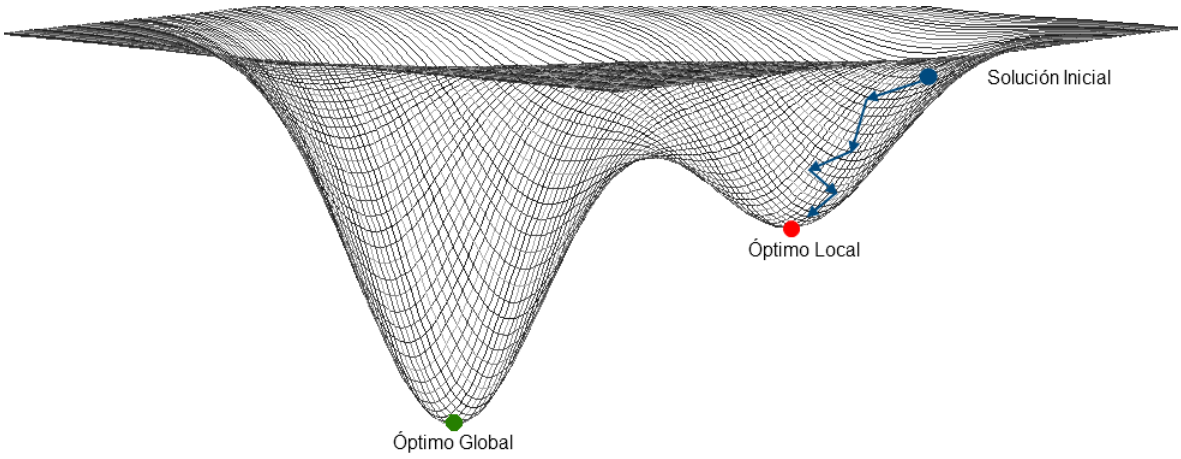

Figura 6.1: Deficiencia de la búsqueda local ordinaria

1..6. Memoria y Clasificaciones Tabu

Una distinción importante en TS tiene que ver con la diferenciación entre memoria de corto plazo y memoria de largo plazo. Cada tipo de memoria tiene asociada su propia estrategia particular. El efecto de ambos tipos de memoria radica en modificar la estructura de vecindario $N(x)$ de solución actual x. La vecindad modificada, que denotamos como $N^*(x)$ es el resultado de mantener una historia selectiva de los distintos estados encontrados durante la búsqueda.

En las memoria de corto plazo, $N^*(x)$ es un subconjunto de $N(x)$ y la clasificación tabu sirve para identificar elementos de $N(x)$ excluídos de $N^*(x)$. En las estrategias para memorias de largo plazo, $N^*(x)$ puede ser extendido para que incluya soluciones que no se encuentran en $N(x)$. Esto implica que la vecindad de una solución x no es un conjunto estático sino un conjunto que puede ir cambiando de acuerdo a la historia que se tiene almacenada.
Mediante el enfoque de memoria a corto plazo, es probable que una solución x sea visitada más de una vez. Un aspecto crucial en TS tiene que ver con la elección apropiada de $N^*(x)$.

Enfoque de Memoria Reciente

Este tipo de memoria selecciona atributos que ocurren en soluciones recientemente visitadas y son etiquetados como atributos *tabu-active*, y las soluciones que contienen este tipo de atributos son consideradas soluciones tabú. Esto previene que ciertas soluciones del pasado reciente pertenecientes a $N^*(x)$ sean visitadas nuevamente. Otras soluciones que compartan tales atributos tampoco serán tenidas en cuenta y no serán visitadas. Los movimientos que aplicados a una solución x llevan a una solución x' con atributos tabu son consideradas movidas tabu.

Definición 6.4. *El tiempo durante el cual un atributo permanece en estado tabu y por consiguiente las soluciones que compartan estos atributos, se denomina tenure.*

El hecho de considerar soluciones tabú a aquellas soluciones que tienen uno o más atributos tabu-active es que potencialmente estaríamos perdiendo la posibilidad de visitar soluciones de muy buena calidad. Hay casos en que este tipo de situaciones no serían deseables. Para evitar este tipo de problemas se introduce el concepto de criterio de aspiración. El criterio de aspiración permite que un movimiento sea admisible aún cuando sea considerado como tabú si el valor objetivo es mejor que la mejor solución conocida hasta el momento.

Memoria de Largo Plazo

Existe también el concepto de memoria a largo plazo basada en frecuencias. Esta memoria almacena las frecuencias de las ocurrencias de atributos en las soluciones visitadas tratando de identificar o diferenciar regiones. En base a estas frecuencias, penaliza o premia movimientos que usan atributos que fueron altamente considerados en el pasado.

1..7. Pseudo-código

A continuación se presenta el algoritmo básico de TS. Este esquema general es el que se siguió para la implementación del algoritmo presentado en esta tesis.

Algorithm 1 Simple Tabu Search();

1: $k \leftarrow 1$;
2: Generar solución inicial;
3: **while** NOT COND PARADA **do**
4: Determinar vecindad $N(x)$
5: Identificar Tabu List $N^*(x, k)$
6: Identificar Lista Aspiración $A(x, k)$
7: Elegir la mejor $x' \in N(x, k) = \{N(x) - N^*(x, k)\} + A(x, k)$
8: Si x' mejora la mejor solución conocida, agrego a lista Tabú
9: $x = x'$
10: $k = k + 1$
11: **end while**

En los siguientes dos capítulos se explicará en detalle la implementación de la metaheurística, en primer lugar explicando el algoritmo de búsqueda de una solución inicial y posteriormente explicando la customización del tabú search a nuestro problema.

Capítulo 7

Heurística RMORN

En este capítulo explicaremos los aspectos principales de la heurística implementada para resolver el problema **RMORN**.

En primer lugar, generalizaremos el proceso de optimización dividiendo en dos bloques principales como puede verse en la Figura 7.1. En el resto del capítulo nos focalizaremos en el algoritmo de construcción de esta solución.

1.. Construcción del Algoritmo de Solución Inicial

El algoritmo toma como entrada todos los datos relacionados al problema y retorna una solución factible inicial. Si bien el Tabú Search no tiene en cuenta si esta solución está cerca del óptimo o no, basado en la experiencia empírica de los casos ejecutados, podemos decir que este algoritmo obtiene una solución más que satisfactoria.
El diagrama en bloques del algoritmo es presentado en la Figura 7.2.

Antes de focalizarnos en cada uno de estos bloques definiremos algunos conceptos que serán empleados a lo largo del capítulo.

Sea un grafo $G = (V, E)$ simple no dirigido entonces consideremos las siguientes definiciones:

Definición 7.1. *G es k-arista-conexo si $|V| > k$ y $G \backslash F$ es conexo para cada conjunto $F \subset E$ que contenga menos de k aristas. El mayor número entero k tal que G es k-arista-conexo se denomina arista-conectividad de G y se representa con $\lambda(G)$. En particular, $\lambda(G) = 0$ si G es disconexo.*

Definición 7.2. *Sea $X \subset V \cup E$. Decimos que X particiona a G si $G \backslash X$ es disconexo. En particular, un vértice que desconecta un grafo G se denomina punto de articulación o punto de corte. Si una arista desconecta al grafo G, esta arista se denomina puente.*

Definición 7.3. *Si $\{V_1, V_2\}$ es una partición de V, el conjunto $E(V_1, V_2)$ de todas las aristas de G atravesando la partición se denomina corte. Un corte mínimo no vacío en G se denomina "bond".*

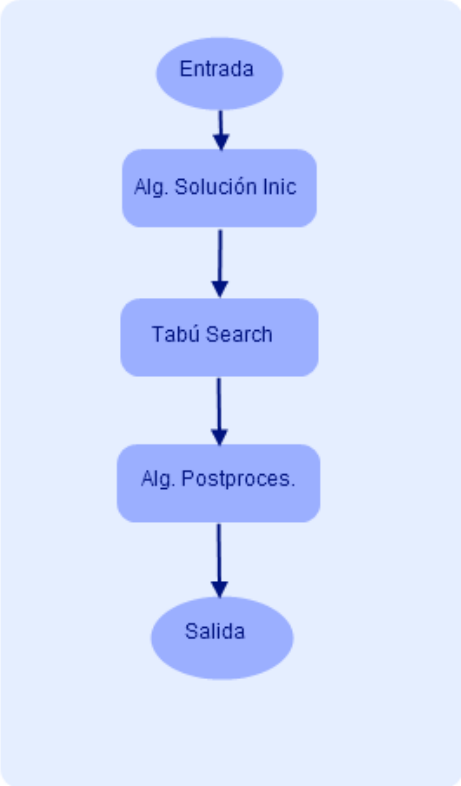

Figura 7.1: Bloque General de la Metaheurística

1..1. Inicialización

Durante la etapa de inicialización, un conjunto de datos son calculados para poder ser utilizados durante la ejecución del algoritmo. Los datos más importantes son:

- Distancia entre nodos.
- Cálculo de bonds estructurales.
- Cálculo de la asociación topológica.

Distancia entre nodos

Los caminos más cortos entre pares de nodos en G_T son calculados utilizando el algoritmo de Dijkstra. Las distancias correspondientes para los nodos v_i, $v_j \in V_D$ son inferidos por los caminos $tns(v_i), tns(v_j) \in V_T$.

Bonds

Sea $G = (V, E)$ un grafo plano. Entonces su *grafo dual* $G^* = (V^*, E^*)$ es tal que existe un vértice v* para cada cara[1] c de G y para cada arista $e \in E$ existe una arista $e^* \in E^*$ tal que los extremos de e^* son los vértices en G^* que vienen de las caras adjacentes a e. La definición de grafo dual no

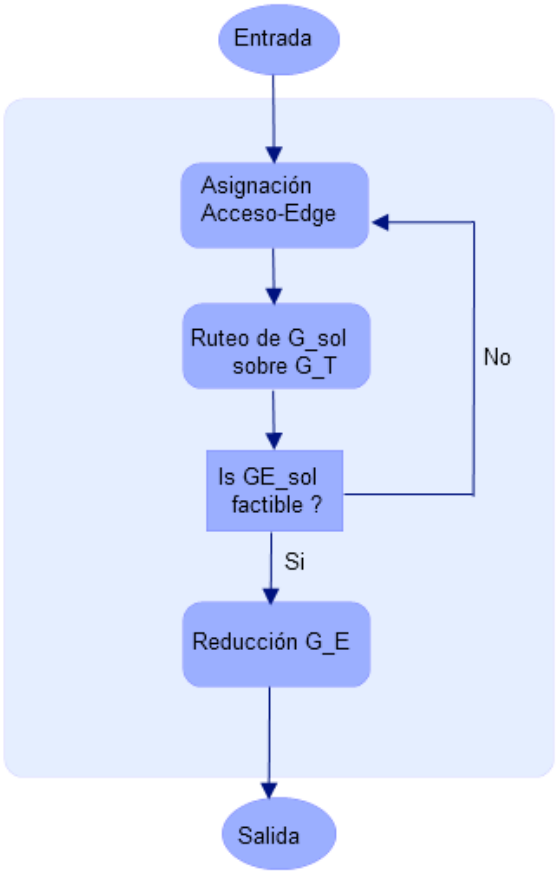

Figura 7.2: Bloque General del Algoritmo Inicial

depende únicamente de la estructura del grafo sino también de la representación plana del mismo. Generalmente, el grafo dual es un multigrafo.
Para nuestro propósito, la representación plana que adoptamos es claramente la actual disposición de la Red de Transporte. En base a esto, necesitamos saber las coordenadas de cada nodo de transporte como dato de entrada a nuestro algoritmo.

El algoritmo que lista los bonds en la Red de Transporte hace uso del grafo dual y de la siguiente propiedad:

[1] Las caras en nuestra topología anillada de transporte sería un anillo.

Nota 7.1. *Un conjunto de aristas en el grafo plano G forma un bond si y solo si sus correspondientes aristas duales forman un ciclo en el grafo dual G^**

Por lo tanto, es suficiente con encontrar todos los ciclos del grafo dual. Sin embargo, aún en un grafo cúbico plano, esta tarea es un problema **NP-Hard** [5]. De modo que no podemos esperar contar con un algoritmo eficiente para listar los cortes y de ahí obtener los bonds. Sin embargo, en nuestro caso, la existencia de pocos nodos de grado mayores a dos nos facilitan el asunto.

En primer lugar, todos los nodos de grado dos son eliminados y el camino que antes conformaban es reemplazado por una arista simple. Obtenemos entonces un nuevo grafo G'_T.
Luego construimos el grafo dual.
En el ejemplo, $G'^*_T = (V'^*_T, E'^*_T)$ donde $V'^*_T = \{A, B, C, D, E, F\}$. Los ciclos elementales son

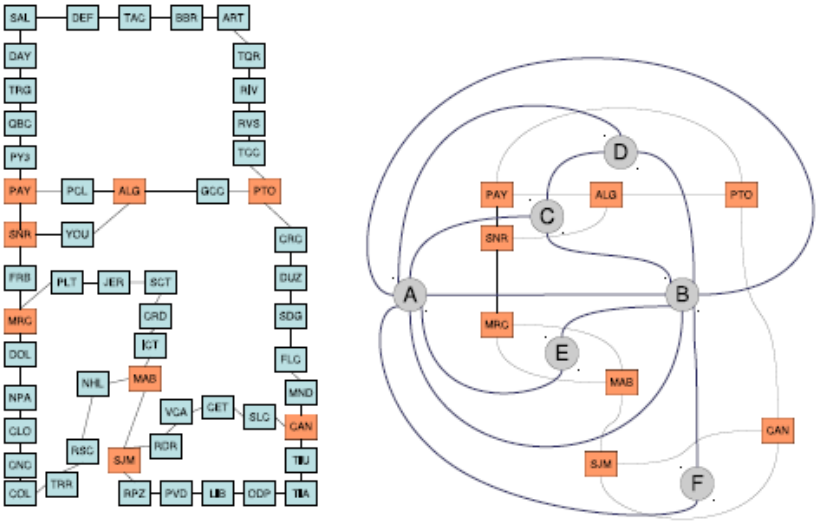

Figura 7.3: Red de Transporte Parte Oeste → Red de Transporte Dual Parte Oeste

aquellos que no contienen ciclos dentro de si mismo. Son definidos por las caras del grafo dual G'^*_T. Sin embargo, una de ellos, por ejemplo la cara externa debe ser dejada de lado pues puede ser obtenida a partir del resto como explicaremos. En el ejemplo vemos que existen ACBA, ACDA, ABEA, CDBC y ADFA. Todos los ciclos del grafo pueden ser obtenidos combinando los grafos elementales. Luego, tenemos a lo sumo $2^5 = 32$ ciclos posibles en el grafo dual. Observamos que a pesar de que ACBA + ACDA = ABCDA es un ciclo válido, ACDA + ABFA = ACDABFA no es un ciclo. De hecho, ACDABFA conforma un "ocho", dos ciclos que comparten un solo vértice.

Una vez que el ciclo es seleccionado, por ejemplo el ciclo ABCDA, el paso final consiste en retornar cual es el bond en el grafo original. La forma de alcanzar esto es la siguiente: para cada arista del ciclo, debemos encontrar su arista correspondiente en el en el grafo original. Por ejemplo, para la arista C-D en el grafo dual tenemos que su correspondiente en el grafo original es PAY-ALG. Debemos recordar que PAY-ALG representa un camino condensado entre estos nodos, donde el camino original es PAY-PCL-ALG. A causa de esto, elegimos una arista a remover de este camino (PAY-PCL ó PCL-ALG). Reptiendo esto para cada arista del ciclo en el grafo dual obtenemos un

bond en el grafo original[2]. Para completar el ejemplo, una combinación particular de aristas de transporte que constituye un bond para el ciclo ABCDA en el grafo dual puede ser: SNR-FRB para AB, YOU-ALG para BC, PAY-PCL para CD y ART-TQR para DA.
El algoritmo es relativamente simple y además puede ser implementado eficientemente utilizando vectores para cada arista.

En vistas de que los bonds son utilizados por la función *Is_$G_E$sol_factible* (ver Figura 7.2) los bonds y los caminos de las aristas de transporte en G_T asociadas a cada arista en $G_T^{'*}$ son precomputadas al inicializar el algoritmo de modo de poder mejorar la performance del mismo.

Asociación Topológica

Del mismo modo que con el cálculo del grafo dual de la sección anterior, este procedimiento crea un nuevo grafo $G_T^{''}$ colapsando los nodos de grado dos en G_T con dos excepciones: un nodo de transporte que comparta una estación con un nodo edge, o un nodo de transporte cuya eliminación genere una multi arista en el grafo resultante. En ambos casos, el nodo queda como está.

Una nueva función distancia se define como $r' : E_T^{''} \rightarrow \mathbb{R}$ tal que tiene en cuenta las distancias de las aristas colapsadas.

El propósito de este proceso es omitir de G_T todas las estaciones y nodos que no cambian la solución encontrada. Esto es así porque una vez que el nodo acceso ha sido asignado a sus respectivos nodos edges, estos no brindarán información a la solución.

1..2. Asignación Acceso-Edge

La asignación Acceso-Edge fue discutida anteriormente. Con el objetivo de repasar la implementación particular de la conexión haremos referencia a [38]. No empleamos este algoritmo directamente en vista de que la asignación Acceso-Edge es también precomputada. Se trabaja con las matrices de demanda $\{\bar{M}_0^{'}, \bar{M}_1^{'}, ..., \bar{M}_{|E_T|}^{'}\}$ que resume el tráfico que intercambia cada nodo edge en cada escenario posible de falla en la Red de Transporte. Estas matrices contemplan el tráfico de los nodos accesos y son agregados a sus correspondientes nodos edges.

1..3. Ruteo de $G_E$sol sobre G_T

La parte principal de este algoritmo intenta rutear los links de G_E a través de la Red de Transporte subyacente de manera balanceada y económica. Para el ruteo balanceado se procede de la siguiente manera. Se genera un grafo colapsado GC como el grafo G_T'' descrito anteriormente. Para cada nodo Edge que tiene tráfico, se sortea un nodo al azar y se enrutan una tras otra las aristas e que tienen a dicho nodo como uno de sus extremos. Una vez enrutada la arista de datos en transporte, se actualiza el costo en GC elevando el costo de las aristas involucradas en el camino en transporte a la potencia de $\frac{3}{2}$.[3]

[2] Muchos bonds pueden ser construidos a partir del mismo ciclo en el grafo dual
[3] Dato calculado empíricamente.

El proceso se repite para cada uno de los nodos edges con tráfico.
El enfoque balanceado tiene que ver con evitar la sobrecarga de links sobre una arista de transporte. Esta situación haría imposible el reruteo de tráfico en el caso de un escenario de falla cuando dicha arista en particular es la que falla. Por lo tanto, es importante que todas las conexiones de un nodo edge hacia sus vecinos sean establecidas de manera balanceada.

1..4. Is $G_E sol$ factible?

El algoritmo *Is_$G_E sol$_factible* es uno de los algoritmos más importantes de la metaheurística aplicada. Se utiliza tanto en el algoritmo de búsqueda de solución inicial así como también en la aplicación del tabú search. Tal como lo indica el nombre, evalúa si una solución es factible o no.

El algoritmo lista los túneles de datos ordenándolos en orden decreciente en el valor del tráfico requerido. Los túneles son enrutados uno por uno a través de los caminos que en ese momento tengan la capacidad necesaria para llevar a cabo la tarea. Más aún, el algoritmo intenta hacer uso de las rutas en G_E con la mínima cantidad de links que sea posible.
Entre rutas con igual cantidad de links, elige aquella con el camino más corto en G_T.
Dado que esta función o algoritmo es llamada muchas veces en cada iteración, se realizaron muchos esfuerzos en hacerla lo más performante posible. Por esta razón, se hace un manejo muy prudente y eficiente de la memoria, optimizando estructuras de datos, almacenando soluciones parciales en caches y precomputando algunos valores que no cambiarán su valor durante la ejecución. El algoritmo actual valida una solución para una red real (tan larga como la red de ANTEL) en pocos minutos, dependiendo del tamaño del caso de prueba que se está corriendo y rechaza una solución no factible, en general, en un tiempo bastante menor.
Más abajo se muestra un pseudocódigo del algoritmo. El algoritmo DijstraD es básicamente el algoritmo de Dijkstra pero tiene en cuenta la capacidad actual de cada tunel. Esto es, si el camino más corto para ir de i a j toma la arista x pero dicha arista no tiene la capacidad suficiente para enrutar el tráfico que se está intentanto enrutar entonces el camino se descarta. Para ello, se simula la remoción de la arista problemática y se lanza el dijkstra nuevamente. El procedimiento se repite hasta que se pudo enrutar, en cuyo caso aquellas aristas que se simuló su remoción se vuelven a integrar, o bien hasta que no exista camino posible para enrutar el tráfico en cuyo caso la solución no será factible.

1..5. Reducción de G_E

El último paso en la etapa de construcción consiste en construir G_E, definiendo capacidades $B : E_E \to \hat{B}$ y túneles de datos $\Phi : (\emptyset \cup E_T) \to 2^{P_D}$, con el ruteo $\Psi : E_E \to 2^{P_T}$ ya determinado. Un esquema del algoritmo de reducción puede verse en la Figura 7.4 donde el objetivo es ir seleccionando bonds, remover los links vinculados a este bond y verificar que la solución sea factible. En caso de ser factible se seguirá iterando sobre el resto de los bonds y en caso contrario se reintegrarán los links, esto es, vuelven a ser considerados como parte de la solución.

Los links que cruzan el bond que se está procesando , esto es, aquellos links de datos que utilizan

Algorithm 2 Is_$G_E sol$_factible;

```
1:  Ordeno demandas decrecientes en tráfico;
2:  Analizo el escenario nominal;
3:  Aplico DijkstraD para cada una de la demandas
4:  if (pude Enrutar todas las demandas) then
5:     Ahora repito para cada escenario de falla;
6:     bool nofactible = false;
7:     efalla = Primer Escenario de Falla;
8:     while (efalla != null AND nofactible == false) do
9:        Aplico DijkstraD para cada una de la demandas y escenario efalla;
10:       if (pude Enrutar) then
11:          Actualizo uso Aristas;
12:          efalla = Siguiente Escenario de Falla // null si no encuentra;
13:       else
14:          nofactible = true;
15:       end if
16:    end while
17:    if (nofactible) then
18:       Return False;
19:    else
20:       Return True;
21:    end if
22: else
23:    No es factible en Escenario Nominal;
24:    Return False;
25: end if
```

alguna de las aristas en transporte que pertenecen al bond en cuestión, son rankeados de acuerdo a su costo en G_T y se procesan primero los más costosos, es decir, se van eliminando en orden decreciente de costo hasta que cierta condición minimal previamente establecida es alcanzada [38]. Si la solución es factible, continuamos con el algoritmo analizando otro bond. Si la solución no es factible se realiza un roll-back, o sea, se vuelven a integrar uno a uno los links en sentido inverso al que fueron removidos hasta que la solución vuelve a ser factible.

De este modo, damos por finalizado el capítulo relativo a la implementación del primer paso a la solución del problema RMORN. En el siguiente capítulo mostraremos en detalle la implementación del Tabú Search para la mejora de la solución de partida.

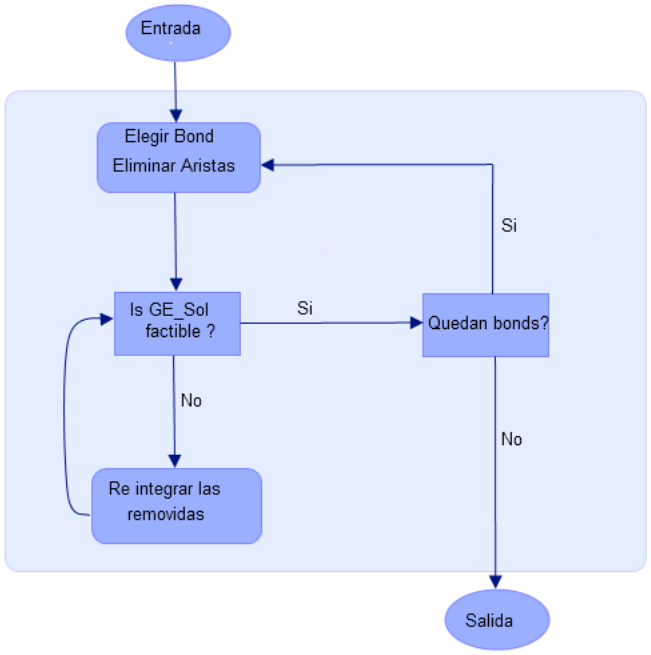

Figura 7.4: Reducción de G_E

Capítulo 8

Customización del Tabú Search

El presente capítulo tiene como objetivo explicar en detalle la customización de la metaheurística Tabú Search para la resolución del problema **RMORN**.
En el capítulo 6 se dio una introducción a las metaheurísticas y se describió a grandes rasgos los principales conceptos en torno a la metaheurística Tabú Search. A continuación se describen las características de la implementación.

1.. Implementación

Una de las cosas que habíamos puntualizado respecto al uso de memoria en TS estaba asociada a si esta era una memoria explícita, en la cual se almacena la solución completa, o si era una memoria de atributos, la cual permite almacenar ciertas características o atributos de soluciones.

Por temas de performance y para hacer más eficiente el algoritmo se optó por implementar una memoria de atributos y no una memoria explícita. En particular, se implementaron dos memorias de atributos, a saber *RemTabuList* y *AddTabuList*.
Las transformaciones que a una solución S se le estarán aplicando para determinar su vecindad $N(S)$ serán la inserción ó la eliminación de una arista.

- *RemTabuList:* Esta memoria mantiene una lista de aquellas aristas que al removerlas de la solución S generaron una solución S' de menor costo que S.
  Es prudente entonces considerar la inserción de estas aristas en la solución como un movimiento tabú por un tiempo[1] determinado.
- *AddTabuList:* Mantiene una lista de aquellas aristas que al agregarlas a la solución S generaron una solución S' de menor costo que S.
  En este caso y por un tenure determinado, el movimiento que remueva una de estas aristas es considerado tabú.

Aquellas aristas de *RemTabuList* que han alcanzado el tenure predefinido serán insertadas en una lista llamada *availableEdges*. Esta lista tiene aquellas aristas que pueden volver a considerarse como

[1] Parámetro *tenure* definido previamente

parte de la solución.
Además, como vimos anteriormente, es probable que el algoritmo de solución inicial devuelva una solución S que haya excluído algunas aristas que formaban parte de los datos de entrada. Estas aristas son mantenidas entonces en la lista *availableList*.

El algoritmo alterna entre movimientos de inserción y remoción. Si hay pocas aristas en *availableList* se privilegian las remociones. En caso contrario, se privilegiarán las inserciones.
Un diagrama de la interacción entre estas listas se muestra en la Figura 8.1

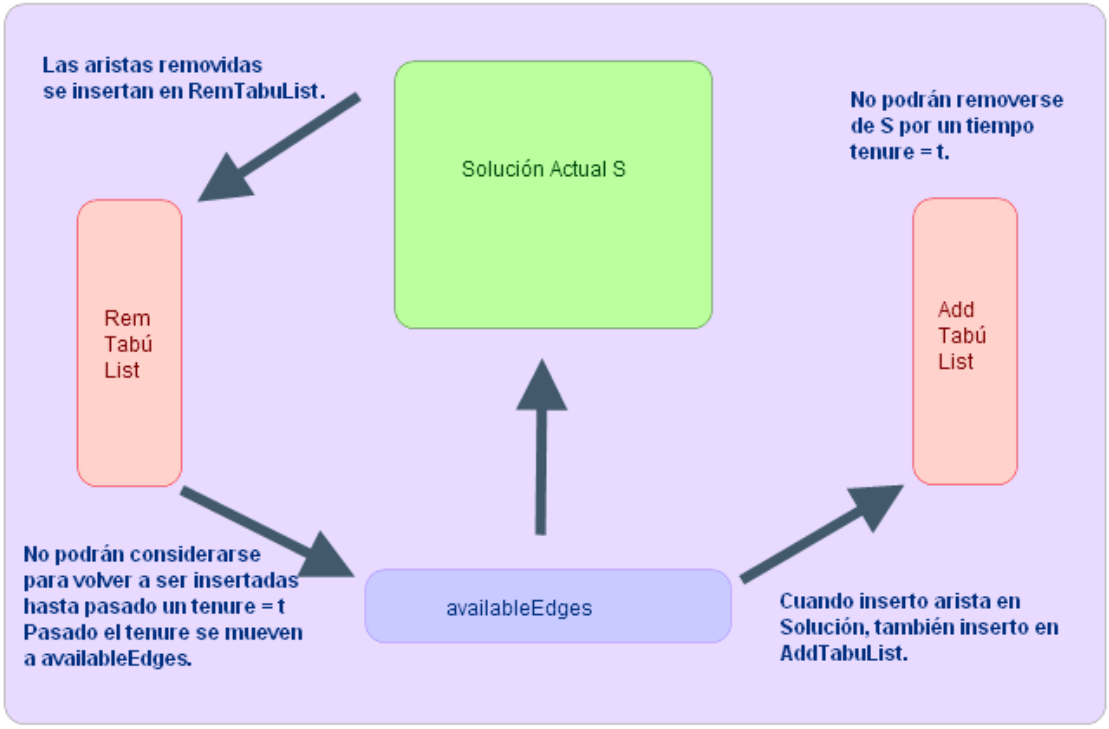

Figura 8.1: Customización del Tabú Search.

De modo empírico y basado en las ejecuciones de los casos de prueba se estableció el tenure en un valor igual a 6 iteraciones.

Como complemento a las listas tabú existe una lista que está asociadas al criterio de aspiración.
A dicha lista la llamamos *bestAttributes* y almacena aquellas aristas que con alta frecuencia aparecen en soluciones de buena calidad.

De este modo, si en determinado momento una de estas aristas se encuentra en la lista *RemTabuList* ocasionalmente dicha arista será removida antes de finalizar su tenure pues es una arista que pertenece a *bestAttributes*. La decisión de remover o no en un tenure menor está estrechamente ligada a la frecuencia de aparición en soluciones de buena calidad. A grandes rasgos, toda vez que una arista forma parte de una buena solución, se incrementa un contador interno para esta arista que contabiliza la frecuencia de aparición en soluciones de buena calidad. Este contador es tenido en cuenta a la hora de purgar aristas de las listas tabú. En caso de que la frecuencia de dicha arista en soluciones de buena calidad sea alta, el algoritmo que purga aristas de las listas tabú la quitará en la mitad de tiempo, es decir, cuando el tenure haya llegado a la mitad (tenure = 3) del valor establecido originalmente (tenure = 6).

Algorithm 3 TabuSearch(SolucionInicial S);

```
1:  k ← 1;
2:  costoActual = costo(S);
3:  bool sale = false;
4:  while (k <maxIter) do
5:     if (availableEdges es vacia) then
6:        ProcesarSinAristasDisponibles();
7:     else
8:        ProcesarConAristasDisponibles(availableEdges);
9:     end if
10:    purgoAristasDeListasTabu();
11:    k = k + 1
12: end while
```

El algoritmo 3 muestra la estructura principal del algoritmo. Por temas de presentación y para hacer más legible el pseudocódigo se dividió el código en dos subrutinas, el algoritmo 4 y el algoritmo 5 que se muestran más abajo.
El algoritmo principal recibe una solución inicial S y consta de un loop que va desde cero a maxIter, invocando alternativamente los algoritmos 4 y 5 dependiendo del estado de la solución y posteriormente purgando las aristas de las listas tabú de ser necesario; esto se logra invocando la función *purgoAristasDeListasTabu*. El proceso anterior se repite para cada iteración.
El algoritmo 4 toma una arista *e* al azar de la solución S que no haya sido visitada e intenta removerla. Si la solución resultante es factible y además el costo de la solución es menor al costo actual entonces dicha arista es agregada a *RemTabuList* y se confirma su remoción de la solución S. En caso de ser factible pero no disminuya el costo o bien, que no sea factible la solución resultante entonces la arista es devuelta a la solución S. Las aristas que se van seleccionando y que no mejoran la solución se van marcando de modo de evitar volver a seleccionarlas. El procedimiento se repite hasta que encuentre una arista que al removerla mejora la solución o hasta que no queden aristas por probar. Una vez alcanzado alguno de estos dos estados se borran las marcas de visita para cada arista afectada.

Para el caso del algoritmo 5, se sortea aleatoriamente que operación se llevará a cabo, si será remoción o bien inserción (esta operación puede hacerse en este caso pues *availableEdges* es no vacía). Si el sorteo determina que la operación sea remoción entonces lo que resta ejecutar será exactamente igual al algoritmo 4.
En caso de que el sorteo determine que la operación es inserción el algoritmo procede de la siguiente

manera: se elije aleatoriamente una arista *e* del vector de aristas disponibles *availableEdges* que no haya sido seleccionada anteriormente e inserto en la solución S. Si la solución resultante es factible y además disminuye el costo de S entonces la arista es agregada a *AddTabuList* y se sale del loop. En caso de que sea factible y no mejore el costo o que no sea factible la arista es devuelta a *availableEdges* y es marcada como arista ya probada para evitar volver a seleccionarse. El procedimiento se repite hasta que encuentre una arista que al ser insertada en la solución disminuya su costo o bien no queden aristas por probar. Una vez alcanzado alguno de estos estados se borran las marcas de visita para cada arista involucrada.

Algorithm 4 ProcesarSinAristasDisponibles();

```
 1: sale = false;
 2: while (!sale) do
 3:    Elijo arista e que no haya probado remover.
 4:    if (Si puedo remover e de solucion) then
 5:       S = S - e;
 6:       factible = esSolucionFactible?(S);
 7:       if (factible == true) then
 8:          nuevoCosto = costo(S);
 9:          if (nuevoCosto < costoActual) then
10:             Agrego arista e a RemTabuList;
11:             sale = true;
12:          else
13:             S = S + e;
14:             Marco e como arista probada;
15:             sale = noHayMasAristas();
16:          end if
17:       else
18:          S = S + e;
19:          Marco e como arista probada;
20:          sale = noHayMasAristas();
21:       end if
22:    else
23:       Marco e como arista testeada;
24:       sale = noHayMasAristas();
25:    end if
26: end while
27: borroProbadas();
```

Algorithm 5 ProcesarConAristasDisponibles(vector availableEdges);

```
sale = false;
while (!sale) do
    Sorteo aleatoriamente operacion: Agregar o Remover.
    if (si operacion Agregar) then
        Elijo arista e de availableEdges que no haya intentado agregar
        S = S + e;
        factible = esSolucionFactible?(S);
        if (factible == true) then
            nuevoCosto = costo(S);
            if (nuevoCosto < costoActual) then
                Agrego arista e a AddTabuList;
                sale = true;
            else
                S = S - e;
                Devuelvo e a availableEdges;
                Marco e como arista probada;
                sale = noHayMasAristas();
            end if
        else
            S = S - e;
            Devuelvo e a availableEdges;
            Marco e como arista probada;
            sale = noHayMasAristas();
        end if
    else
        Elijo arista e que no haya probado remover.
        if (Si puedo remover e de solucion) then
            S = S - e;
            factible = esSolucionFactible?(S);
            if (factible == true) then
                nuevoCosto = costo(S);
                if (nuevoCosto < costoActual) then
                    Agrego arista e a RemTabuList;
                else
                    S = S + e;
                    Marco e como arista probada;
                    sale = noHayMasAristas();
                end if
            else
                S = S + e;
                Marco e como arista probada;
                sale = noHayMasAristas();
            end if
        else
            Marco e como arista testeada;
            sale = noHayMasAristas();
        end if
    end if
end while
borroProbadas();
```

El algoritmo *purgoAristasdeListaTabu* recorre cada una de las listas tabú en busca de aquellas aristas que puedan ser removidas de dichas listas. Las aristas a remover serán aquellas que hayan alcanzado el tenure especificado o bien aquellas que cumplan con el criterio de aspiración anteriormente comentado.

2.. Algoritmo de Postprocesamiento

Este algoritmo tiene como objetivo buscar aquellas aristas que pueden ser dimensionadas con una capacidad menor a la que fue dimensionada en los algoritmos previos y que mantengan la factibilidad de la solución. Para ello, el algoritmo elige la primer arista e y si existe una capacidad inferior a la capacidad con la que fue dimensionada entonces el algoritmo asigna esta capacacidad y corre el algoritmo *Is_*G_E*sol_factible*. Si el resultado es una solución factible entonces la capacidad de dicha arista es cambiada al nuevo valor, en caso contrario se deja la dimensión anterior.
El proceso se repite para cada una de las aristas.
Luego de este post procesamiento es de esperar que el costo de la solución se haya reducido sustancialmente.

Cerramos de esta forma la descripción en detalle de la implementación de la solución al problema. En los sucesivos capítulos se presentarán los resultados y conclusiones obtenidos durante la ejecución del algoritmo Tabú Search.

Parte IV

Resultados.

Capítulo 9

Descripción de los Casos de Prueba

Los casos de prueba introducidos en este capítulo fueron creados por el equipo de proyecto con el objetivo de evaluar el trabajo realizado y presentar los resultados de manera consistente. Se introdujeron algunas variantes hechas por el equipo a los casos de prueba de ANTEL de modo de poder plantear diferentes situaciones. Estas variantes son presentadas en detalle en este capítulo.

1.. Set de Escenarios 1

1..1. Demanda

La forma de la red está principalmente definida por el tráfico residencial de Internet. Otro tipo de tráficos tales como "Tráfico Empresarial", "Tráfico de valor agregado" (IPTV, VoIP, Cardales) ha demostrado ser menos significativo. El tráfico residencial de Internet es dominante en el sentido de que la presencia o la ausencia de los otros tráficos prácticamente no afecta a G_E, o si lo hace, será de manera insignificante [38].

La razón de este bajo impacto es la cantidad relativamente baja de clientes o la baja demanda de ancho de banda en el caso de los servicios a empresas.

Los parámetros identificados como importantes a la hora de realizar los test de performance tienen que ver con los clientes y el ancho de banda, es decir, la cantidad de servicios vendidos y el consumo asociado de ancho de banda de cada uno de los servicios.

Consideraremos dos tipos de casos de prueba que toman en cuenta diferentes demandas sobre la Red y que serán llamados *demL* y *demH*. El primer caso representa un escenario de baja demanda y el segundo un escenario de alta demanda. Ambos casos son extremos pero son escenarios plausibles en el contexto de ANTEL.

1..2. Requerimientos

Además de los requerimientos de demanda, la forma de la Red resultante depende del consumo de productos ofertados por las empresas de telecomunicaciones. En particular, en una variedad de productos libres y flexibles.

Este concepto fue modelado como $z_Q(bw)$[38]. Por tanto, otra variable que tendremos en cuenta a la hora de crear los casos de prueba y analizar sus resultados es la función z_Q.

Llamaremos *zQL* y *zQH* a los escenarios con baja z_Q y con alta z_Q respectivamente.

1..3. Contenido

El costo de utilización de los enlaces internacionales juega un rol importante cuando el contenido demandado por los clientes se encuentra en el exterior dado que ANTEL debe pagar a compañías en el exterior para acceder a dicho contenido. Por otro lado, si el contenido requerido por los clientes es nacional, entonces el costo será inferior.

Como resultado, una estrategia seguida para intentar reducir el costo de estos enlaces internacionales es la utilización de "caches". De este modo, contenido que originalmente se encontraba en el extranjero es almacenado localmente y puede ser accedido a un costo inferior.
Los escenarios creados reflejan el porcentaje del contenido nacional demandado. Las dos alternativas manejadas son denotadas como *intL* e *intH*. El primero contiene un 25 % de tráfico local (nacional) mientras que el segundo, en su totalidad, es tráfico internacional.

1..4. Arquitectura

Por último, creemos importante explorar otras formas de arquitecturas de red, distintas a la actualmente manejada. En la actualidad, las Redes de Agregación ATM e IP/MPLS tienen por objetivo fundamental concentrar tráfico HSI hacia otra red; conocida internamente como "Red IP Pública". Conceptualmente, la "Red IP Pública" es la porción "Uruguay" de Internet. Tiene presencia física en cuatro centrales, Aguada, Unión, Centro y Cordón, además del NAP de las Américas en Miami. Centraliza todos los enlaces internacionales y es la responsable de rutear el tráfico a Internet.

Con la anterior Red ATM no existían alternativas al respecto. Pero la nueva Red de Agregación MPLS presenta como "rasgo potencial", el de cumplir ambas funciones: concentración / ruteo. Por lo tanto, resulta interesante evaluar cuantitativamente si la fusión de ambas funciones representa algún beneficio económico.

Definimos dos variables: *napL* y *napH* para representar respectivamente que: se mantiene la arquitectura actual o se extiende la presencia de la Red de Agregación IP/MPLS hasta los puntos de intercambio internacional de tráfico, siempre teniendo en cuenta ambas funciones: concentración y ruteo.

2.. Datos del Problema

Como se dijo anteriormente, todos los datos fueron sumistrados por ANTEL. La Red de Transporte G_T (nodos, aristas, distancias), capacidades disponibles $\hat{B}$ y su costo por kilómetro $T : \hat{B} \to \mathbb{R}_0^+$. La Red de Datos ha sido brindada mayoritariamente por ANTEL y se han realizado modificaciones para mantener concordancia con algunas abstracciones del modelo y escenarios. La función z_Q y las demandas fueron derivadas por el equipo de proyecto [38].

La red de Transporte utilizada como parámetro de entrada para los algoritmos se muestra en la Figura 9.1 La Red de Datos que consiste de nodos y aristas de datos también fue brindada por AN-

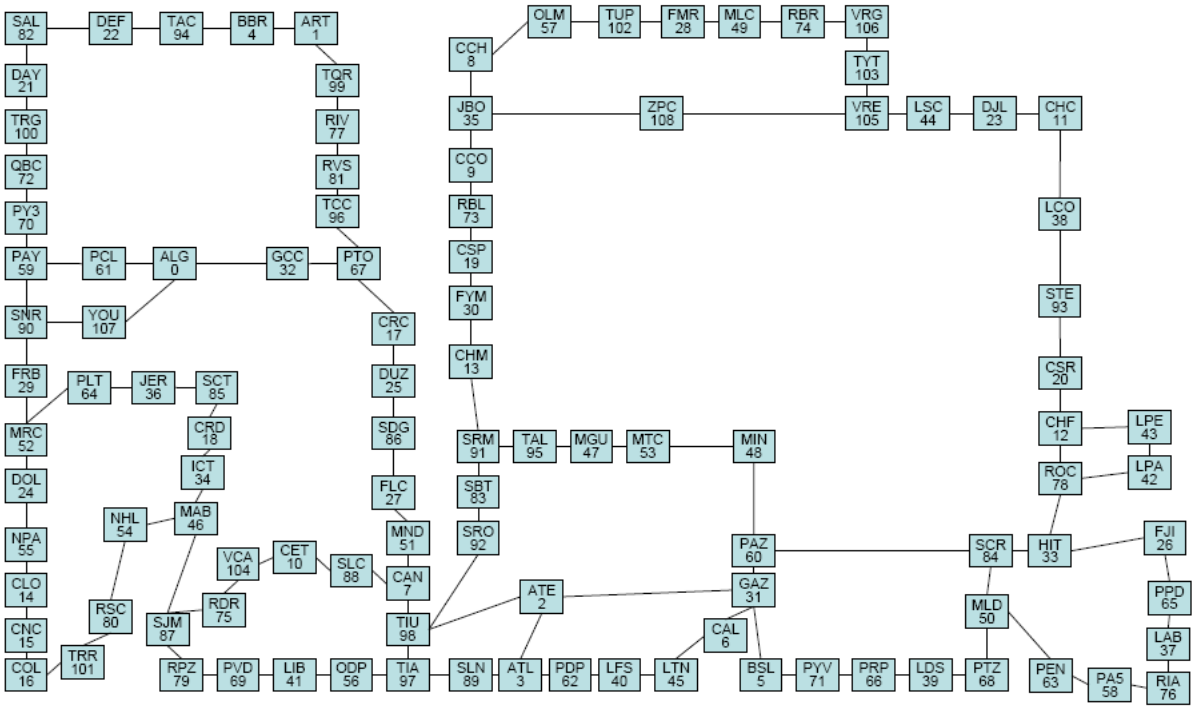

Figura 9.1: Red de Transporte ANTEL.

TEL. Los nodos de acceso fueron establecidos en casi todas las estaciones de red donde el servicio ADSL es ofrecido sobre la infraestructura ATM. Para los nodos edges se manejaron varias situaciones. Se han incluído nodos fijos para representar AMM y/o Internet en los casos que correspondían[1], pero todo el resto de los nodos son opcionales.

Entre estos figuran todas las capitales o ciudades importantes, todos los nodos cuya estación tiene en la Red de Transporte grado tres o mayor.

En la Tabla 9.1 el conjunto $\hat{B}$ de velocidades disponibles y sus costos es presentada.

Ya hemos explicado el significado de *intL*, *intH* así como también *napL* y *napH*. Combinando *dem*, *zQ*, *int* y *nap* construimos 16 casos de prueba diferentes para analizar los cuales son resumidos en

[1] En aquellos escenarios que Internet es un nodo de la Red de Agregación, donde termina el tráfico internacional.

$\hat{B}$	**Velocidad**	**Mapeo VCAT/DWDM**	**Capacidad Útil**(Mbps)	**Equivalente E1**	**Costo**(US$/km)
b_0	0 Mbps	-	0 Mbps	0 E1	0
b_1	10 Mbps	5 VC12	9.5 Mbps	5 E1	3.46
b_2	20 Mbps	10 VC12	19 Mbps	10 E1	6.92
b_3	40 Mbps	20 VC12	38 Mbps	20 E1	13.84
b_4	50 Mbps	1 VC3	42 Mbps	21 E1	14.54
b_5	100 Mbps	2 VC3	84 Mbps	42 E1	29.10
b_6	140 Mbps	1 VC4	132 Mbps	63 E1	43.60
b_7	280 Mbps	2 VC4	264 Mbps	128 E1	88.60
b_8	560 Mbps	4 VC4	528 Mbps	256 E1	177.20
b_9	1000 Mbps	7 VC4	924 Mbps	441 E1	305.25
b_{10}	10000 Mbps	1 λ	10000 Mbps	5263 E1	104.00

Tabla 9.1: Capacidades y Costos

la Tabla 9.2.

Nº	**Escenario de Prueba**
1	demH_zQH_intH_napH
2	demH_zQH_intH_napL
3	demH_zQH_intL_napH
4	demH_zQH_intL_napL
5	demH_zQL_intH_napH
6	demH_zQL_intH_napL
7	demH_zQL_intL_napH
8	demH_zQL_intL_napL
9	demL_zQH_intH_napH
10	demL_zQH_intH_napL
11	demL_zQH_intL_napH
12	demL_zQH_intL_napL
13	demL_zQL_intH_napH
14	demL_zQL_intH_napL
15	demL_zQL_intL_napH
16	demL_zQL_intL_napL

Tabla 9.2: Set de Escenarios 1

En la Figura 9.2 se ilustra para el caso de prueba 02 (02-demH_zQH_intH_napL), la topología dela Red de Transporte, los nodos de datos y las potenciales aristas de datos a incluír en la solución.

3.. Set de Escenarios 2

Con el objetivo de testear la performance con otro set de parámetros, hemos creado un conjunto nuevo de casos de prueba. Con este propósito, hemos dividido la red original de Transporte en dos

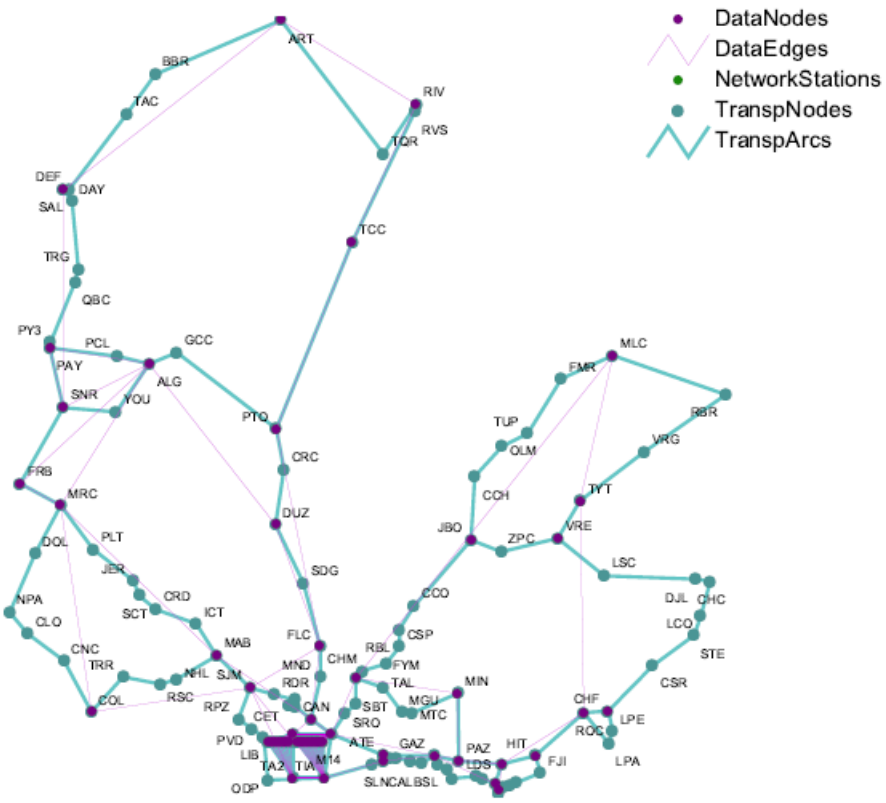

Figura 9.2: Caso de Prueba 02.

regiones, a saber Región Este y Región Oeste, como se muestra en la Figuras 9.3 y 9.4 manteniendo la topología de anillo y los nodos TIU y TIA en ambas infraestructuras. Hemos dejado los nodos de datos de cada región y hemos inducido aristas de datos. Además, hemos introducido modificaciones. Por ejemplo, hemos creado una nueva gama de bandwidths que van desde 1000 a 10000 Mbps aumentando de a 1000 Mbps y con costo proporcional al ancho de banda, esto se muestra en la Tabla 9.3. Estos nuevos valores resultan útiles para el algoritmo de post procesamiento que se ejecuta luego del tabú search y que tiene como proposito reducir al mínimo la capacidad de dimensionamiento de las aristas siempre y cuando se mantenga la factibilidad.

Por otro lado, hemos creado casos de prueba en los cuales hemos modificado completamente el grafo de datos probando con topologías "full-mesh" *fm* y "half-full-mesh" *hfm*. Para el último esquema, hemos removido la mitad de la totalidad de aristas de datos del grafo completo. En este caso, esperamos que la introducción de potenciales túneles de datos también significará una disminución en el costo. Por último, para alguno de los nuevos escenarios, hemos cambiado su demanda original de tráfico. En comparación con el estado inicial, hemos incrementado la cantidad de tráfico en la red. Para ello, hemos asignado randómicamente tráfico uniformemente distribuído desde 0 a 30000 Mbps a pares de nodos. El número de nodos con demanda de tráfico es la mitad del número de aristas de datos. Para estos casos de prueba hemos anexado la palabra *charge*, dando la noción de que en estas redes, casi todas las aristas serán diseñadas y cargadas, de ser factible, con un valor próximo a su máxima capacidad.

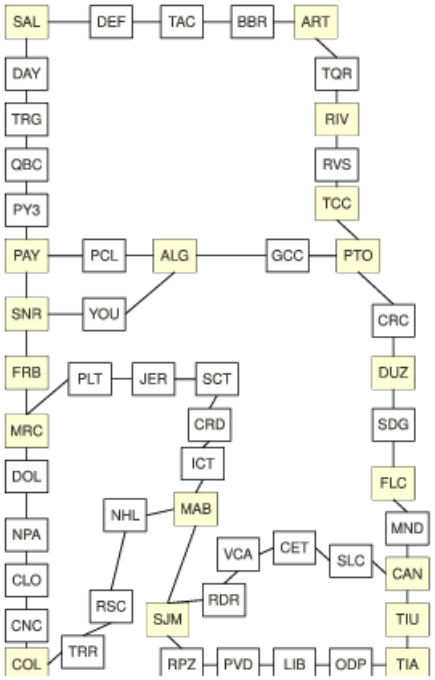

Figura 9.3: Región Este.

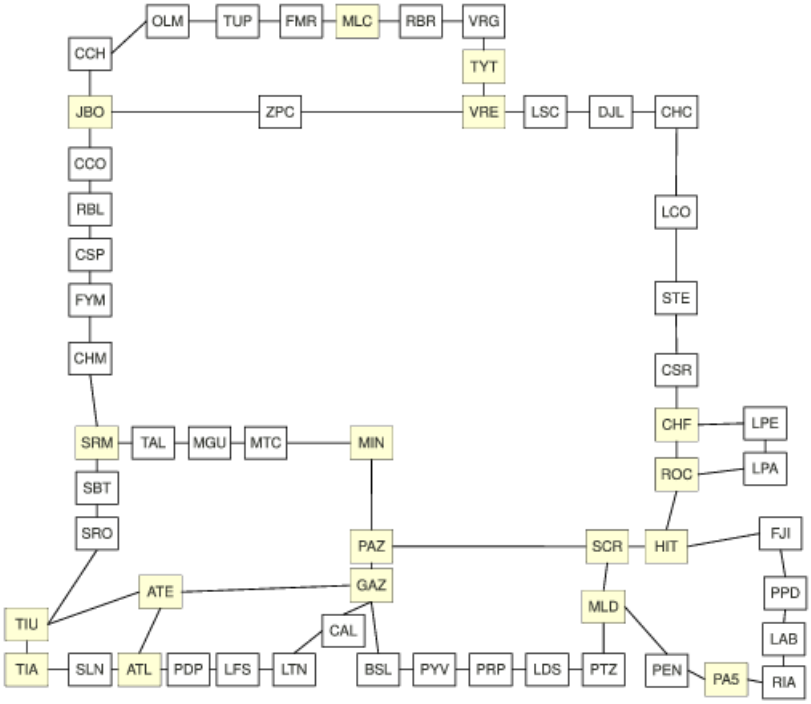

Figura 9.4: Región Oeste.

$\hat{\mathbf{B}}$	**Velocidad**	**Costo**(US$/km)
b_0	0 Mbps	0
b_1	1000 Mbps	10
b_2	2000 Mbps	20
b_3	3000 Mbps	30
b_4	4000 Mbps	40
b_5	5000 Mbps	50
b_6	6000 Mbps	60
b_7	7000 Mbps	70
b_8	8000 Mbps	80
b_9	9000 Mbps	90
b_{10}	10000 Mbps	100

Tabla 9.3: Capacidades y Costos de la Red de Transporte

En el siguiente capítulo se mostrarán los resultados obtenidos luego de haber corrido el algoritmo con los distintos escenarios de prueba.

Capítulo 10

Análisis de Resultados

1.. Parámetros del testing

En este capítulo se presentarán los resultados experimentales obtenidos al aplicar el algoritmo implementado con el esquema descrito previamente.

Los resultados obtenidos para el primer set de datos, Set de Escenarios 1, son similares a aquellos presentados en [38]. Allí, los resultados son presentados desde un punto de vista económico, analizando los beneficios y las estrategias que ANTEL podría seguir. Este análisis, números, datos, resultados, recomendaciones ya fueron elevados a la contraparte de ANTEL.

En este trabajo, nos focalizamos más que nada en el análisis computacional de los datos, generando casos de prueba semi aleatorios que se corresponden con el Set de Escenarios 2.

Un punto importante fue la elección del lenguaje de programación para la implementación de la solución. Esto fue una decisión fundamental ya que muchos requerimientos no funcionales tales como performance de los algoritmos, manejo de memoria, etcétera suelen estar afectados por el lenguaje de programación que se utilice y obviamente, que tan bien se programa. Nuestro problema requiere que el manejo de grandes cantidades de datos e información así como el de recursos sean manejados de manera lo más eficientemente posible. También, por afinidad en el tipo de programación habitual de los integrantes del equipo así como también por las bondades que nos podría brindar se pensó en un lenguaje de programación orientado a objetos.
De este modo, se decidió implementar la solución en $C++$ pudiendose compilar y correr tanto en *Windows* como en *Linux*. Los resultados que se incluyen en esta tesis fueron obtenidos en un equipo AMD Athlon Dual Core Processor de 1.9 GHz y 2 Mbytes de memoria RAM corriendo sobre *Windows XP*.

Los parámetros definidos para la ejecución son dos, *MaxIter* el cual contabiliza la cantidad de iteraciones que el algoritmo emplea hasta retornar un resultado cuyo valor fue especificado en 100 y *Tenure* que forma parte del TS y ya fue explicado anteriormente cuyo valor fue especificado en 6.

Los parámetros que se utilizarán para mostrar los resultados son:

- Gap entre el algoritmo de solución inicial y el tabú search para un problema p y que se define como:

$$gap(p) = \frac{SI - TSC}{SI}$$

  donde *SI* y *TSC* representan el costo de la solución inicial y el costo luego de ejecutar el TS respectivamente.

- Costo Solución Inicial.
- Costo Solución Final.
- Tiempo total *T* de ejecución.

2.. Resultados para Set de Escenarios 2

Para este set de casos de prueba el número de iteraciones fue establecido en una etapa de tuning previa con el valor *MaxIter = 100*. La Tabla 10.1 resume la información obtenida para los parámetros especificados.

Un resultado notorio es la capacidad del tabú search para, a partir de una solución inicial, lograr optimizar en las sucesivas iteraciones el costo de la solución final. Esto puede verse en los porcentajes del *gap* que se muestran en la tabla. Además, los tiempos de ejecución resultaron súmamente razonables para la dimensión de los casos de prueba.

Como puede verse, algunas de las redes no resultaron factibles, lo cual es coherente por la carga que fue agregada a cada caso de prueba. Un resultado interesante es que estos fueron escenarios que contenían la mitad de las aristas de datos posibles a considerar. Otro aspecto positivo surge del análisis del tiempo de ejecución para los casos en que la solución no era factible. Como puede verse, el algoritmo rápidamente detecta la no factibilidad para estos casos particulares.

Entrando en detalles, se puede decir que en general, cuanto mayor es la cantidad de aristas que pueden elegirse mejor resultados en el *gap* se obtienen. Esto puede confirmarse analizando los resultados de los casos de prueba *copy* con unas pocas aristas potenciales de datos versus los casos de prueba full-mesh *fm* con todas las aristas disponibles para considerar.

También se puede ver que los resultados en los casos de prueba *cap* son mejores que el resto, tal cual se esperaba.

Caso de Prueba	Factible?	Costo Inicial	Costo Final	gap %	T(seg)
east_copy	Si	128911	112207	13	90
east_copy_cap	Si	123953	107891	13	77
east_copy_charge_cap	Si	111247	95185	14	140
east_fm	Si	358957	114536	68	51
east_fm_cap	Si	358957	112207	69	49
east_fm_charge	Si	2364476	1662190	30	152
east_fm_charge_cap	Si	2273534	1598260	30	468
east_hfm	No	-	-	-	1.6
east_hfm_cap	No	-	-	-	1.1
east_hfm_charge	No	-	-	-	1.6
east_hfm_charge_cap	No	-	-	-	1.6
west_copy	Si	1753369	977650	44	78
west_copy_cap	Si	1685932	940048	44	112
west_copy_charge_cap	Si	1685932	940048	44	40
west_fm	Si	807412	460764	43	67
west_fm_cap	Si	776358	340656	56	43
west_fm_charge	Si	6256829	4540690	27	133
west_fm_charge_cap	Si	6016181	4366050	27	131
west_hfm	No	-	-	-	0.5
west_hfm_cap	No	-	-	-	1.7
west_hfm_charge	No	-	-	-	1.2
west_hfm_charge_cap	No	-	-	-	2.3

Tabla 10.1: Set de Escenarios 2

Capítulo 11

Conclusiones

1.. Conclusiones de la Solución Implementada

En primer lugar creemos que dado el alcance y la complejidad del problema, la solución presentada en este trabajo cumple los objetivos y expectativas trazados al comienzo de este trabajo. En el presente trabajo se desarrolló un algoritmo basado en la metaheurística tabú search que logra soluciones de buena calidad en un tiempo prudente para redes reales.

Durante el desarrollo de este trabajo se buscó siempre la eficiencia en los algoritmos implementados tratando siempre de realizar un buen manejo de las estructuras de datos y memoria así como también la optimización de los métodos auxiliares. Sin embargo, dadas las características de este problema fue inevitable el uso de grandes estructuras de datos para mantener en memoria redes de datos y redes de transporte, entre otras cosas. De este modo, pensamos que más allá del esfuerzo nuestro en pos de la eficiencia, aún podríamos mejorar algunos algoritmos para hacerlos más ágiles a la hora de ejecutarlos usando cachés que almacenen resultados intermedios y de este modo evitar recalcular resultados.

Para concluir, remarcamos que esta tesis resuelve un problema real complejo en el cual se desarrolló una herramienta que puede ser utilizada para mejorar la toma de decisiones y planificación estratégica para una entidad como ANTEL. Esta herramienta podría ayudar a identificar problemas estructurales en la red y poder simular una solución alternativa a tal problema. Además, puede ser útil para determinar la estrategia de negocios más económica a seguir así como también justificar inversiones objetivamente y medir el impacto que estas puedan tener. Podría servir también para evaluar la robustez y el costo de una modificación en la infraestructura actual o bien para dimensionar o planificar el ruteo sobre la Red de Transporte. Por último, otra posible utilización de esta herramienta podría ser la evaluación cuantitativa del costo de expandir los servicios de ANTEL a cada punto del país.

2.. Trabajo a Futuro

Como se comentó en la sección anterior, siempre podrá trabajarse en los métodos, algoritmos y estructuras de datos de modo de poder hacerlos más rápidos y eficientes a la hora de ejecutarse.

Si bien la herramienta desarrollada brinda la capacidad de obtener una gama de resultados útiles para la empresa ANTEL no existen requerimientos de presentación de estos resultados, es decir, los datos que se presentan son básicamente tablas con resultados y, en el caso del Set de Escenarios 1, gráficas que complementan dicha información y presentan de un modo más amigable estos datos. Una extensión que podría hacerse podría ser la implementación de una interfaz gráfica en la cual se muestre un mapa con las distintas redes, se brinde la capacidad de cargar la información de input y poder visualizar gráfica y dinámicamente el dimensionamiento y ruteo de las demandas de tráfico que se cargan inicialmente. Esta interfaz podría desarrollarse de modo de poder interactuar con el algoritmo pudiendo detener la ejecución, tomar decisiones "on the fly" como el redimensionamiento de cierta arista y continuar la ejecución. Creemos que una interfaz de este estilo podría llegar a ser de gran utilidad y potenciaría en gran medida la herramienta ya desarrollada.

Este proyecto ha inspirado la realización de múltiples trabajos en paralelo aplicando diferentes técnicas de resolución basadas en otras metaheurísticas tales como VNS [35–37], Algoritmos Genéticos [33, 34], Grasp [29], y Métodos Exactos para pequeñas instancias. Como trabajo a futuro podría estudiarse la posibilidad de poder resolver el problema basados en alguna otra metaheurística adecuada para este tipo de problemas tales como Simulated Annealing, Algoritmos de Enjambres, entre otros, a efectos comparativos.

Por último, en este trabajo hemos creado un conjunto de casos de prueba ajenos a los casos originales brindados por la empresa ANTEL de modo de poder evaluar mejor la performance de nuestros algoritmos. Una amplia gama de casos de prueba, aparte de los nuestros, pueden ser generados y probados usando esta herramienta de modo de poder tomar decisiones cruciales y estrategias convenientes a futuro.

Bibliografía

[1] Dimitris Alevras, Martin Grötschel, and Roland Wessäly. A network dimensioning tool. In *Preprint SC 96-49, Konrad-Zuse-Zentrum fur Informationstechnik*, 1996.

[2] Yossi Azar and Oded Regev. Combinatorial algorithms for the unsplittable flow problem. *Algorithmica*, 44:49–66, 2006.

[3] Daniel Bienstock, Sunil Chopra, Oktay Günlük, and Chih-Yang Tsai. Minimum cost capacity installation for multicommodity network flows. *MATHEMATICAL PROGRAMMING*, 81:177–199, 1998.

[4] B. Fortz and M. Poss. An improved benders decomposition applied to a multi-layer network design problem. *Operations Research Letters*, 37(5):359 – 364, 2009.

[5] Johnson D. Garey, M. *Computers and intractability: a guide to the theory of NP-completeness.* W. H.Freeman and Company, 1979.

[6] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[7] G. N. Frederickson and J. Jájá. *Approximation algorithms for several graph augmentation problems*. Number 10. 1981.

[8] S. Khuller and R. Thurimella. *Approximation algorithms for graph augmentation*. Number 14. 1993.

[9] Alexander Gersht and Alexander Shulman. A new algorithm for the solution of the minimum cost multicommodity flow problem. In *Decision and Control, 1987. 26th IEEE Conference on*, volume 26, pages 748 –758, 1987.

[10] Raja Jothi, Balaji Raghavachari, and Subramanian Varadarajan. A 5/4-approximation algorithm for minimum 2-edge-connectivity. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 725–734, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[11] H. Kerivin, D. Nace, and T.-T.-L. Pham. Design of capacitated survivable networks with a single facility. *Networking, IEEE/ACM Transactions on*, 13(2):248 – 261, 2005.

[12] Hervé Kerivin and A. Ridha Mahjoub. Design of survivable networks: A survey. In *In Networks*, pages 1–21, 2005.

[13] Sunggy Koo, G. Sahin, and S. Subramaniam. Cost efficient lsp protection in ip/mpls-over-wdm overlay networks. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 2, pages 1278 – 1282 vol.2, 2003.

[14] Zheng Ma, Jiang Chen, Yang Richard Yang, and Arvind Krishnamurthy. Optimal capacity sharing of networks with multiple overlays. In *In Proceedings of IEEE IWQOS*, 2006.

[15] Thomas L. Magnanti, Prakash Mirchandani, and Rita Vachani. Modeling and solving the capacitated network loading problem. *Operations Research Center Working Paper*, 239(91), 1991.

[16] Thomas L. Magnanti, Prakash Mirchandani, and Rita Vachani. Modeling and solving the two-facility capacitated network loading problem. *Operations Research*, 43(1):pp. 142–157, 1995.

[17] M.Stoer. Design of survivable networks. In *Lecture Notes in Mathematics*, volume 1531. Springer-Verlag, Berlin, 1992.

[18] S. Orlowski, A. M. C. A. Koster, Christian Raack, and Roland Wessäly. Two-layer network design by branch-and-cut featuring mip-based heuristics. In *Proceedings of the Third International Network Optimization Conference (INOC 2007), Spa, Belgium*, April 2007.

[19] Papadimitriou C.H. Itai, A. and J.L. Szwarcfiter. *Hamilton paths in grid graphs*, volume 11. New York, NY, USA, 1982.

[20] D. Peleg and E. Reshef. *Deterministic polylog approximation for minimum communication spanning trees*. Springer-Verlag.

[21] Lancia G. Bafna V. Chao K. Ravi R. Wu, B. Y. and C. Y. Tang. *A polynomial time approximation scheme for Minimum routing cost spanning tree*. 1998.

[22] Christian Raack, Arie M.C.A. Koster, Sebastian Orlowski, and Roland Wessäly. On cut-based inequalities for capacitated network design polyhedra. *Networks*, 2010.

[23] S. Ramamurthy and B. Mukherjee. Survivable wdm mesh networks. ii. restoration. In *Communications, 1999. ICC '99. 1999 IEEE International Conference on*, volume 3, pages 2023 –2030 vol.3, 1999.

[24] S. Ramamurthy and B. Mukherjee. Survivable wdm mesh networks. part i-protection. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 744 –751 vol.2, March 1999.

[25] L. Sahasrabuddhe, S. Ramamurthy, and B. Mukherjee. Fault management in ip-over-wdm networks: Wdm protection versus ip restoration. *Selected Areas in Communications, IEEE Journal on*, 20(1):21 –33, January 2002.

[26] M. Skutella. Approximating the single source unsplittable min-cost flow problem. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 136 –145, 2000.

[27] Xingwei Wang, Lei Guo, Fei Yang, Tengfei Wu, and Wei Ji. Multi-layer survivable routing mechanism in gmpls based optical networks. *Journal of Systems and Software*, 81(11):2014 – 2023, 2008.

[28] Richard T. Wong. A survey of network design problems. *Operations Research Center Working Paper*, 080(78), 1978.

[29] Paola Festa, Mauricio, and G. C. Resende. Grasp: An annotated bibliography. In *Essays and Surveys in Metaheuristics*, pages 325–367. Kluwer Academic Publishers, 2002.

[30] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. Bradford Company, Scituate, MA, USA, 2004.

[31] P. J. M. Laarhoven and E. H. L. Aarts, editors. *Simulated annealing: theory and applications*. Kluwer Academic Publishers, Norwell, MA, USA, 1987.

[32] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[33] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, USA, 1975.

[34] S. N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2007.

[35] P. Hansen and N. Mladenović. Variable neighborhood search. In P. Pardalos and M. Resende, editors, *Handbook of Applied Optimization*, pages 221–234. Oxford University Press, 2002.

[36] P. Hansen and N. Mladenović. Tutorial on variable neighborhood search. Technical Report G–2003–46, Les Cahiers du GERAD, 2003.

[37] P. Hansen and N. Mladenović. Variable neighborhood search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 145–184. Kluwer Academic Publisher, 2003.

[38] Claudio Risso. Optimización de costos en redes multicapa robustas. Master's thesis, Facultad de Ingeniería, UdelaR, http://premat.fing.edu.uy/IngenieriaMatematica/archivos/tesis_claudio_risso.pdf, 2010.