ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

ANÁLISIS DE RUTAS DINÁMICAS DEL SIMULADOR SUMO EN REDES VEHICULARES MEDIANTE MÉTRICAS DE CENTRALIDAD DE GRAFOS USANDO LA LIBRERÍA IGRAPH DE R. CASO DE ESTUDIO: SECTOR LA CAROLINA (QUITO)

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

LUIS FELIPE ZUMÁRRAGA CADENA

DIRECTOR: PhD. LUIS FELIPE URQUIZA AGUIAR

Quito, marzo de 2022

AVAL

Certifico que el presente trabajo fue desarrollado por Luis Felipe Zumárraga Cadena, bajo mi supervisión.

PhD. LUIS FELIPE URQUIZA AGUIAR
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo Luis Felipe Zumárraga Cadena, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

_____

LUIS FELIPE ZUMÁRRAGA CADENA

DEDICATORIA

A mi madre, por ser el pilar fundamental de mi vida y que me motiva a seguir adelante a pesar todos los obstáculos y por todo el esfuerzo y sacrificios que ha hecho para poder brindarme siempre lo mejor.

A mi familia, por confiar siempre en mí, ser mi apoyo en los momentos más difíciles y demostrarme que no existen imposibles cuando se encomiendan las cosas a Dios.

AGRADECIMIENTO

A Dios, por la salud y la vida, por poner a tantas personas buenas en mi camino, que de una u otra forma me han ayudado perseguir mis sueños.

A mi madre y mi familia, por su apoyo incondicional e impulsarme a cumplir cada una de las metas que me he propuesto.

A mis amigos, por todos los momentos que hemos compartido y nos han servido para crecer profesional y personalmente.

Un agradecimiento especial al PhD. Luis Felipe Urquiza, por su paciencia y ayuda en el desarrollo de este trabajo de titulación y todo su cariño durante el tiempo compartido en las aulas de clases.

ÍNDICE DE CONTENIDO

RESUMEN

En este trabajo se presenta un análisis de las rutas dinámicas generadas en el simulador SUMO que permiten que los vehículos cambien su ruta en función de las condiciones del tráfico frente a rutas estáticas que son establecidas previo a la ejecución de la simulación. Para esto se analizan algunas métricas de los grafos generados a partir de los archivos de salida de las simulaciones mediante el uso de la librería igraph disponible para el lenguaje R.

En el capítulo 1 se describen las principales características de SUMO, sus componentes y la función de reenrutamiento de los vehículos durante la simulación. Además, se estudian los conceptos básicos de la teoría de grafos y las características del lenguaje R, especialmente la librería igraph que permite trabajar con grafos de una manera simple.

En el capítulo 2 se muestra la depuración del mapa obtenido a partir de una herramienta externa OSM, la configuración de los archivos necesarios para la simulación de cada uno de los escenarios propuestos y el procesamiento de los archivos de salida de la simulación.

En el capítulo 3 se obtienen valores de las métricas para los casos simulados y en base a estos se determinan los efectos de usar las rutas dinámicas en lugar de las rutas estáticas que se usan por defecto en el simulador.

Finalmente, en el capítulo 4 se presentan las conclusiones y recomendaciones que derivan de la realización de este proyecto de titulación.

PALABRAS CLAVE: SUMO, Reenrutamiento, Grafos, Métricas, Lenguaje R, Igraph

ABSTRACT

This project presents an analysis of the dynamic routes generated in the SUMO simulator that allow vehicles to change their route depending on the traffic conditions against static routes that are established before to the execution of the simulation. For this, some metrics of the graphs generated from the output files of the simulations are analyzed by using the igraph library available for the R language.

Chapter 1 describes the main features of SUMO, its components and the rerouting function of vehicles during the simulation. In addition, the basic concepts of graph theory and the characteristics of the R language are studied, especially the igraph library that allows working with graphs in a simple way.

Chapter 2 shows the debugging of the map obtained from an external tool OSM, the configuration of the files necessary for the simulation of each of the proposed scenarios, and the processing of the simulation output files.

In chapter 3, values of the metrics are obtained for the simulated cases and based on these the effects of using the dynamic routes instead of the static routes that are used by default in the simulator are determined.

Finally, chapter 4 presents the conclusions and recommendations derived from the completion of this degree project.

KEYWORDS: SUMO, Rerouting, Graphs, Metrics, R Language, Igraph

1. INTRODUCCIÓN

El tráfico vehicular es una de las aristas más importantes que se toma en cuenta al momento de diseñar ciudades sustentables, ecológicas y eficientes, por lo tanto, uno de los mayores problemas es la congestión, que se refleja en largos tiempos de viaje, mayor consumo de combustible y un mayor porcentaje de contaminación [1].

De acuerdo con el Anuario de Estadísticas de Transporte 2018 (ANET) desarrollado por el Instituto Nacional de Estadísticas y Censos (INEC), en ese año se contabilizaron 2'237.264 vehículos en el Ecuador, de los cuales el 22,5% se encuentran Pichincha, por lo que esta se convierte en la provincia con la mayor cantidad de vehículos. De manera específica, Quito registra un promedio de 174 vehículos matriculados por cada 1000 habitantes [2].

Dado este problema del tráfico vehicular, tanto el sector industrial como la comunidad científica han impulsado el uso de las Tecnologías de Información y Comunicación (TICs) para diseñar sistemas de movilidad más eficientes y todos estos desarrollos han sido adoptados dentro del concepto de las ciudades inteligentes o *Smart Cities* [3], [4]. Una parte fundamental para el diseño de estas aplicaciones son las herramientas de simulación que permiten probar diferentes procesos y sistemas complejos de distintas áreas antes de implementarlos.

Los simuladores son herramientas de software que permiten recrear el comportamiento de cualquier fenómeno a partir de datos conocidos acerca del mismo, por lo que son ampliamente utilizados en distintos campos de aplicación [5]. De forma específica, para el transporte, los simuladores permiten determinar los problemas y fallos de los sistemas de movilidad vehicular actuales, evaluar los efectos de posibles cambios o simplemente desarrollar pruebas sobre estos sistemas. Por esta razón, el uso de los simuladores aporta al análisis de soluciones de los problemas relacionados con la congestión vehicular.

Existen varios simuladores de tráfico vehicular y sistemas de transporte dependiendo de la necesidad, sin embargo, uno de los más conocidos y utilizados en proyectos de gran relevancia es el Simulador Urbano de Movilidad (SUMO) [6], que consiste en un conjunto de herramientas informáticas que permiten simular tráfico sobre redes vehiculares o mapas de varias fuentes. Una de las ventajas de este simulador es que dispone de un generador de tráfico que permite configurar el número de vehículos que se usarán en las posibles rutas a seguir, utilizando para esto algoritmos de enrutamiento bien conocidos como Dijkstra [7].

Sin embargo, para la mayoría de las simulaciones en SUMO se utilizan modelos de movilidad que calculan las rutas para los vehículos en función del tiempo y la distancia, pero lo hacen tomando en cuenta un escenario vacío. Por esta razón, durante la simulación los vehículos podrían experimentar retardos considerables mientras que hay rutas alternativas, comúnmente mucho menos congestionadas, que tendrían un tiempo de llegada a destino menor. Este hecho provoca que las trazas de movimientos vehiculares no sigan un comportamiento realista. En consecuencia, los resultados que se pudieran obtener con esas trazas no serían confiables.

Para mitigar este problema, en el presente trabajo se pretende utilizar la capacidad de SUMO de reenrutar los vehículos durante la simulación de manera que ahora se usen rutas menos congestionadas cuando la ruta originalmente planeada se congestione. Se realizará un análisis de los grafos obtenidos a partir de los reportes de SUMO para determinar los cambios en la topología de red vehicular al utilizar la característica de reenrutamiento en el simulador SUMO comparado con el uso de rutas estáticas a lo largo de toda la simulación.

1.1. OBJETIVOS

El objetivo general de este Proyecto Técnico es: analizar las rutas dinámicas del simulador SUMO en redes vehiculares mediante métricas de centralidad de grafos usando la librería igraph de R. Caso de Estudio: Sector La Carolina (Quito).

Los objetivos específicos del Proyecto Técnico son:

- Analizar las principales características del software de simulación SUMO, la teoría de grafos y el lenguaje R.
- Describir la herramienta de Reenrutamiento de vehículos en SUMO.
- Simular diferentes casos de densidad vehicular y tipo de rutas con SUMO.
- Obtener los valores de las métricas de los diferentes casos de simulación.
- Comparar los resultados obtenidos de las simulaciones realizadas.

1.2. ALCANCE

El presente trabajo consistirá en comparar las trazas generadas con rutas estáticas en relación con las generadas en base a rutas dinámicas, cuando se permite el reenrutamiento tomando en cuenta la congestión en la ruta original, de manera que estos datos se utilicen como insumo para la simulación de redes vehiculares.

Se estudiarán las principales características del simulador SUMO, en especial aquellas herramientas que permiten generar rutas dinámicas y la librería igraph de R que permite trabajar con grafos.

Para este trabajo se toma como caso de estudio el sector La Carolina, en el norte de la ciudad de Quito. Este escenario tendrá los siguientes límites: al norte hasta la Av. Gaspar de Villarroel, al sur hasta la Av. Francisco de Orellana, al este hasta la Av. 6 de diciembre y Av. Eloy Alfaro y al oeste hasta la Av. América y este será obtenido de OSM *(Open Street Map)*. En este escenario se generarán tres casos de simulación con diferentes densidades vehiculares, tomando las horas de mayor, menor y congestión media. El tráfico total generado se basará en los porcentajes de vehículos matriculados del Ecuador y en los datos proporcionados por la Secretaría de Movilidad del Distrito Metropolitano de Quito.

Los archivos de salida obtenidos de las simulaciones realizadas serán procesados mediante un filtro que permite establecer las conexiones que han tenido los vehículos durante su trayecto en la simulación que generan a su vez archivos que serán utilizados como entrada en R. Para medir el posible impacto de disponer de rutas dinámicas sobre los resultados de las simulaciones de redes vehiculares se analizará la topología de la red en diferentes instantes de tiempo (por ejemplo, cada 20 segundos). La topología se obtendrá considerando la ubicación de los vehículos en cada instante de tiempo y un rango de cobertura estimado.

Cada topología será analizada mediante las métricas de centralidad de la teoría de grafos. Se usarán al menos las siguientes métricas: (1) Número de nodos, (2) Grado de los nodos, (3) Número de comunidades, (4) Diámetro, (5) Distancia media, (6) Transitividad, (7) Densidad de aristas, (8) Centralidad basada en grado, (9) Centralidad basada en intermediación.

Una vez que se han obtenido los valores de las métricas se puede realizar una comparación entre el comportamiento de las simulaciones usando solo rutas estáticas y rutas dinámicas.

1.3. MARCO TEÓRICO

Las simulaciones de tráfico facilitan la evaluación de los cambios en la infraestructura vial, así como de nuevas políticas antes que estas sean implementadas en la vida cotidiana [8]. Por ejemplo, gracias a estas simulaciones se pueden probar y optimizar los algoritmos para el cambio de luces en los semáforos o determinar nuevas rutas que permitan descongestionar las calles durante las horas pico.

Se pueden aplicar distintos análisis a los resultados obtenidos en estas simulaciones para aprovechar al máximo estos conjuntos de datos. Una de estas herramientas son los grafos que permiten observar las posiciones de los vehículos en un instante dado y analizar las conexiones formadas entre ellos.

A continuación, se presenta una pequeña descripción de cada una de las herramientas que serán utilizadas en este trabajo de titulación, desde la simulación del escenario escogido hasta la obtención de los valores de las principales métricas en los grafos generados en base a los datos a la salida de estas simulaciones.

1.3.1. SUMO (SIMULATION OF URBAN MOBILITY)

SUMO es un paquete de simulación vehicular muy potente que con el paso de los años ha ido añadiendo una gran variedad de herramientas que permiten disponer de escenarios y condiciones de simulación cada vez más realistas [7].

Este software es desarrollado y mantenido por el Instituto de Sistemas de Transporte del Centro Aeroespacial Alemán (DLR) y ha sido diseñado para simular redes viales de gran tamaño como una ciudad completa y obtener como salida archivos con datos acerca de las emisiones de cada vehículo, la cantidad de combustible utilizado por cada vehículo, su tiempo de viaje, entre otros parámetros que pueden ser muy útiles para optimizar los sistemas de movilidad y desarrollar soluciones en materia de tráfico vehicular.

1.3.1.1. Descripción del simulador

El Simulador Urbano de Movilidad (SUMO, por sus siglas en inglés) es un paquete de simulación de código abierto, multimodal, microscópico, discreto en tiempo y continuo en tiempo [9]. Multimodal se refiere a la posibilidad de simular diferentes clases de vehículos como autos particulares, transporte público, motocicletas, bicicletas y hasta peatones.

Se considera un simulador de tráfico microscópico porque permite modelar el comportamiento de cada vehículo de forma independiente, es decir, se puede asignar a cada auto una ruta propia y este se mueve de forma independiente, a diferencia de simuladores de tráfico macroscópico, donde el componente básico es el flujo de vehículos.

Al necesitar un mayor nivel de detalle para las simulaciones, SUMO requiere de un tiempo de cálculo más extenso que otras herramientas. En la Figura 1 se puede observar los distintos tipos de tráfico soportados por los simuladores.

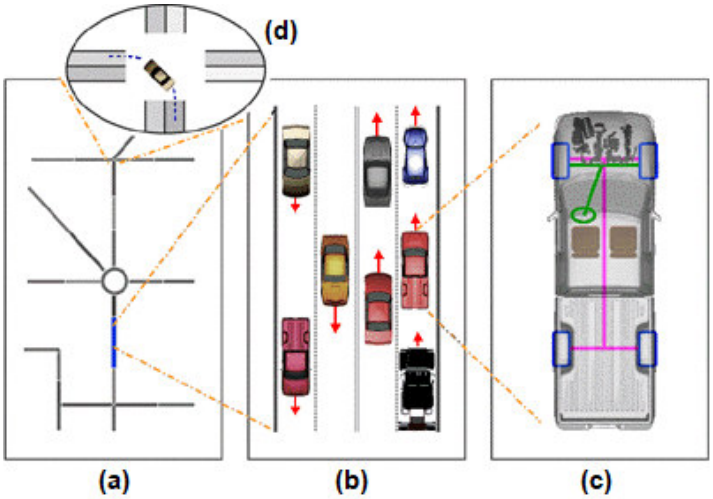

Figura 1.1. Tipos de tráfico soportados por los simuladores: (a) Macroscópico, (b) Microscópico, (c) Submicroscópico, (d) Mesoscópico. Tomado de [10]

Este tipo de simulaciones son las que se prefieren cuando se analizan las redes formadas por estos vehículos, ya que se puede medir de mejor manera los parámetros de los protocolos de comunicaciones y las aplicaciones relacionadas y se pueden obtener patrones de movilidad e información del tráfico más precisa [4].

La simulación es continua en el espacio porque el vehículo puede encontrarse en cualquier posición de la vía, que se describe mediante un número de tipo flotante. A su vez, se trata de una simulación discreta en tiempo porque la simulación se divide en períodos de tiempo y los saltos van de un intervalo a otro.

1.3.1.2. Funcionamiento del simulador

Dependiendo de los requerimientos de cada simulación, el proceso y configuración del escenario pueden llegar a ser muy sencillos, puesto que SUMO incluye una serie de herramientas destinadas a cada una de las necesidades. La primera fase es la generación de la red vial que puede realizarse de dos formas distintas: mediante la creación de una red sintética (por ejemplo, una red en forma de malla o telaraña) o a través de la importación de mapas reales desde repositorios externos [11].

La segunda etapa consiste en la depuración manual del escenario elegido. Esto se realiza con el fin de eliminar inconsistencias entre la realidad y los datos importados en el mapa de la fase anterior. En este punto también se deben realizar validaciones como los giros permitidos o los tiempos de los semáforos. Para esto SUMO incluye algunas herramientas como el editor gráfico Netedit.

La tercera fase abarca la generación de la demanda de tráfico, es decir, se deben especificar las características de los vehículos presentes en la simulación (tipo, dimensiones, etc.) y además se debe definir la ruta que seguirán estos autos durante la simulación. Las rutas pueden ser asignadas a un solo vehículo o un conjunto de vehículos, denominado flujo [12]. Para esta tarea se disponen de varias opciones: especificar una ruta de forma manual, indicando cada una de las calles que conforman la ruta. Una segunda opción es generar rutas de forma aleatoria o también se pueden usar las herramientas propias de SUMO para el cálculo de rutas que basan su funcionamiento en diferentes paradigmas de enrutamiento.

Luego, se deben definir archivos adicionales que pueden resultar necesarios para la simulación y que generalmente están relacionados con la infraestructura de la red vial como los programas de semáforos, sensores, paradas de bus, entre otros. Dependiendo de cada simulación, estos archivos pueden ser omitidos.

Una buena práctica consiste en desarrollar un archivo de configuración de la simulación escrito en formato XML *(Extensible Markup Language)* en el que se pueden indicar los archivos de entrada y opciones de procesamiento, enrutamiento y otras configuraciones que dependerán de los requerimientos de cada simulación.

Con todos los archivos preparados, se puede ejecutar la simulación, ya sea mediante consola o una interfaz gráfica que permite visualizar el avance de la simulación y, por último, se pueden obtener archivos XML con diferentes parámetros de la simulación como los tiempos de viaje de cada vehículo, las emisiones de gases, etc. Estos datos pueden ser aprovechados en análisis posteriores para determinar comportamientos o establecer tendencias que permitan optimizar otras simulaciones o proyectos.

1.3.1.3. Generación de rutas con DUAROUTER

Entre las diferentes herramientas disponibles en SUMO para la generación de la demanda de tráfico, una de las más importantes es DUAROUTER, que permite asignar rutas tanto a viajes individuales como a flujos de vehículos. Para esto utiliza el algoritmo de enrutamiento de la ruta más corta Dijkstra, aunque se pueden utilizar otros como AStar o CHWrapper [13].

Esta herramienta recibe como parámetros de entrada la red vial y el archivo de demanda que puede contener elementos del tipo *trip* para viajes individuales o tipo *flow* para flujos de vehículos. Como salida de se obtiene un archivo con la ruta genera de forma automática para cada uno de los vehículos indicados [11].

En el Código 1.1 se muestra un archivo de configuración de DUAROUTER en el que se especifican los archivos de entrada, el nombre del archivo que se genera como salida y algunas de las opciones de procesamiento. En este caso, se utiliza el algoritmo Dijkstra para el enrutamiento, la opción *randomize-flows* para asignar tiempos de partida aleatorios y la bandera *ignore-errors* para descartar rutas que no se pueden construir. Otras opciones de procesamiento se pueden revisar en [13].

```
<configuration>
    <input>
        <net-file value="mapa.net.xml"/>
        <route-files value="flows.xml"/>
        <additional-files value="busStop.xml"/>
    </input>
    <output>
        <output-file value="autos.rou.xml"/>
    </output>
    <processing>
        <routing-algorithm value="dijkstra"/>
        <randomize-flows value="true"/>
        <ignore-errors value="true"/>
    </processing>
</configuration>
```

Código 1.1. Ejemplo de configuración para DUAROUTER

1.3.1.4. Reenrutamiento de vehículos

Una parte esencial durante la simulación es el tipo de rutas que los vehículos seguirán, ya que estos pueden mantener la ruta establecida en la etapa de generación de demanda o pueden cambiar su ruta en función de parámetros como la situación actual del tráfico o cuando alguna de las calles está bloqueada. Este proceso de asignar una ruta al vehículo o conjunto de vehículos se conoce como enrutamiento.

1.3.1.4.1. Enrutamiento estático

Consiste en la forma más simple en la que un vehículo cumple con su ruta, pues este mantiene la ruta que se ha establecido previo a la simulación, sin importar las condiciones del tráfico. No obstante, al existir congestionamiento, los tiempos de viaje pueden incrementarse y, por lo tanto, algunos vehículos experimentarán grandes retrasos para llegar a su destino.

Estas rutas son generadas con herramientas como DUAROUTER o de forma manual. Sin embargo, no siempre resulta útil utilizar este tipo de rutas y se introduce el concepto de

reenrutamiento, que permite a los vehículos tomar una ruta diferente para llegar a su destino. Esto puede ser útil en situaciones en las que los cambios en la red vial se dan durante el tiempo de ejecución o simplemente se necesita simular este comportamiento en los vehículos para recrear escenarios de congestión en la vida real.

A continuación, se describe el mecanismo que SUMO dispone para habilitar el reenrutamiento de los vehículos durante la simulación y que se utilizará a lo largo de este trabajo.

1.3.1.4.2. Reenrutamiento dinámico

Este enfoque de reenrutamiento permite proporcionar a varios o todos los vehículos la capacidad de volver a calcular su ruta periódicamente. Para esto se toma en cuenta el estado actual del tráfico en la red, es decir, se adapta a posibles colas de vehículos formando congestión y otros cambios dentro de la red.

Para configurar este tipo de reenrutamiento se disponen de las siguientes opciones para indicar cuáles vehículos estarán equipados con el dispositivo de reenrutamiento, con qué frecuencia se tomarán las decisiones de reenrutamiento y la manera en que se calcularán los tiempos de viaje a partir del estado actual de la red.

Tabla 1.1. Opciones de configuración del reenrutamiento dinámico

Opción	Default	Descripción
--device.rerouting.probability *Tipo float*	-1	La probabilidad de que un vehículo tenga un dispositivo de reenrutamiento.
--device.rerouting.explicit *Tipo string*		Asignar un dispositivo de reenrutamiento a vehículos nombrados.
--device.rerouting.deterministic *Tipo Boolean*	False	Los dispositivos se establecen de forma determinística usando una fracción de 1000.
--device.rerouting.period *Tipo string*	0	Período con el cual los vehículos deciden si deben o no calcular una nueva ruta.
--device.rerouting.pre-period *Tipo string*	60	Período de reenrutamiento antes de la incersión o salida de la red.

En adición a estas opciones, existen otras que en este trabajo no se han utilizado ya que no corresponden con la simulación que se va a realizar. Por ejemplo, se podría configurar un archivo de pesos iniciales de cada una de las calles o *edges,* pero no se dispone de la información necesaria para generar este archivo inicial. Otra opción puede ser el uso de

archivo de distritos, no obstante, por la herramienta elegida para la generación de la demanda de tráfico (DUAROUTER), no se disponen de archivos de distritos sino solamente archivos en los que se indican los orígenes para los diferentes flujos de vehículos.

Las opciones de configuración indicadas en la Tabla 1.1 deben ser incluidas en del archivo de configuración de la simulación dentro de una etiqueta *routing,* tal como se muestra en el ejemplo del

```
<configuration>
    <!-- Más opciones de configuración -->
    <routing>
        <device.rerouting.probability value="0.25"/>
        <device.rerouting.period value="60"/>
        <device.rerouting.pre-period value="300"/>
        <device.rerouting.deterministic value="false"/>
        <device.rerouting.explicit value="auto"/>
    </routing>
    <!-- Más opciones de configuración -->
</configuration>
```

Código **1.2.** Ejemplo de configuración del reenrutamiento dinámico

en el que han omitido el resto de las etiquetas que permiten definir los archivos de entrada, salida y las opciones de procesamiento.


```
<configuration>
    <!-- Más opciones de configuración -->
    <routing>
        <device.rerouting.probability value="0.25"/>
        <device.rerouting.period value="60"/>
        <device.rerouting.pre-period value="300"/>
        <device.rerouting.deterministic value="false"/>
        <device.rerouting.explicit value="auto"/>
    </routing>
    <!-- Más opciones de configuración -->
</configuration>
```


Código 1.2. Ejemplo de configuración del reenrutamiento dinámico

1.3.2. AWK

AWK es un lenguaje de programación diseñado principalmente para el procesamiento de archivos de texto, que permite procesar estos datos con pocas líneas de código, por lo que es una herramienta bastante utilizada en el análisis de datos. Cuando se utiliza en minúsculas (awk), se refiere al comando Unix que interpreta los programas escritos en este lenguaje [14], [15].

El gran poder y capacidad de AWK está en su forma de procesar los archivos de entrada, ya que lee cada línea como si fuera un registro, separa este registro en campos y hace lo que se le ordene con esos campos para producir un flujo como salida. Para ejecutar un programa, basta con indicar un archivo de órdenes (programa) y un archivo de entrada que contiene los datos [16].

En el Código 1.3 se muestra la sintaxis básica de una línea que forma parte de un programa AWK, donde *patrón* corresponde a una expresión regular y *acción* es una orden que debe ejecutarse, como. AWK lee el archivo de entrada, y cuando encuentra una línea que coincide con el *patrón*, ejecuta las órdenes especificadas en *acción*

```
/patrón/ {acción}
```

Código 1.3. Sintaxis básica en AWK

Por lo general, los programas incluyen las formas *begin,* que ejecuta las acciones al iniciar la ejecución, antes de que los datos sean procesados y *end,* para ejecutar ciertas órdenes al final de la ejecución después del procesamiento de los datos de entrada. En el código, se observa un programa de ejemplo escrito en AWK.

```
BEGIN { FS="[^a-zA-Z]+"}
 { for (i=1; i<=NF; i++)
      words[tolower($i)]++
 }
END { for (i in words)
     print i, words[i]
 }
```

Código 1.4. Ejemplo de un programa escrito en AWK

1.3.3. TEORÍA DE GRAFOS

Cuando se trabaja con redes de comunicación, como el caso de las redes vehiculares, no es necesario que todas las estaciones puedan comunicarse directamente con el resto de dispositivos, ya que pueden actuar como un puente para pasar un mensaje hacia otra estación [17].

En este contexto, no interesa la posición de las estaciones sino la conectividad entre ellas, y así surge la idea de un grafo, que consiste en un grupo de nodos con una o varias conexiones que se conocen como aristas. Una arista conecta dos nodos o en casos especiales, un nodo consigo mismo [18].

1.3.3.1. Conceptos principales

Un grafo es una terna $G = (V, M, P)$, donde V representa el conjunto de elementos del grafo, denominado vértices o nodos. M es el conjunto de aristas y P es la función que permite asociar a cada elemento de M, un par de elementos del conjunto V [17].

Dicho de otra forma, un grafo consiste en dos conjuntos finitos de nodos (V) y aristas (M) que se relacionan mediante la función P. Considerando un grafo G, sus nodos se denotan como G_V, las aristas G_M y la función de aristas G_P.

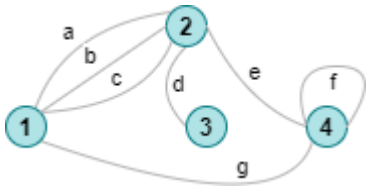

Figura 1.2. Representación de un grafo

La representación de un grafo se presenta en la Figura 1.2, donde las circunferencias representan los nodos y las líneas que conectan estas circunferencias son las aristas. Además, se puede notar que las conexiones pueden ser entre dos nodos o de un nodo consigo mismo.

Los grafos pueden clasificarse en distintos tipos dependiendo de la forma en la que se defina la relación entre los elementos. Las principales clasificaciones de los grafos se presentan a continuación.

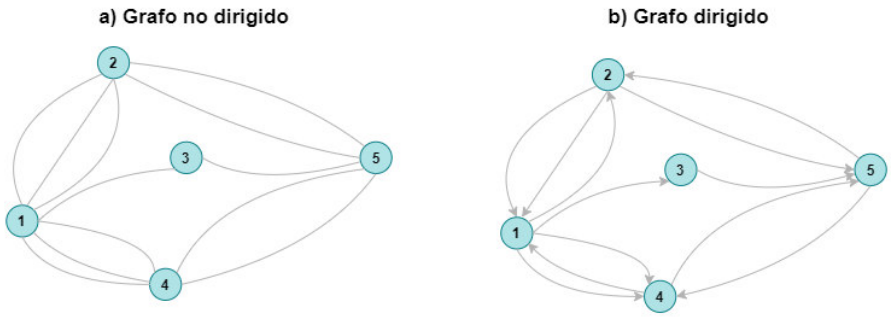

Figura 1.3. Clasificación de grafos: a) No dirigido, b) Dirigido

Cuando las aristas no indican el sentido en el que se produce la conexión, es decir, no se muestran las flechas que representan el sentido de la comunicación, se trata de un grafo no dirigido. En caso contrario, se trata de un grafo dirigido.

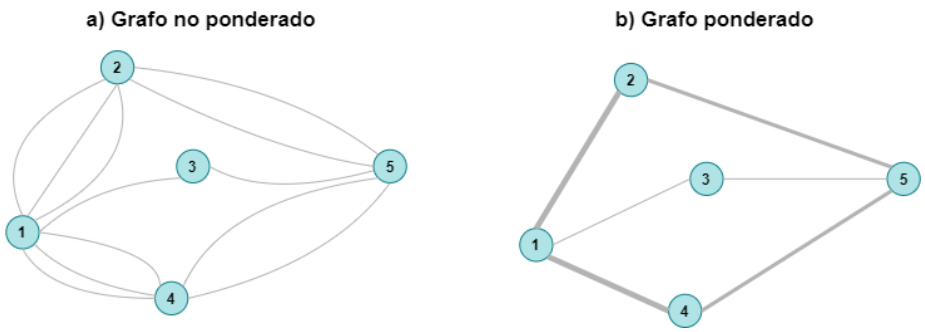

Figura 1.3. Clasificación de grafos: a) No ponderado, b) Ponderado

Cuando se conectan los nodos con una sola línea que tiene asociada alguna característica que revele el número de conexiones entre ese par de nodos, se trata de un grafo ponderado o etiquetado. En cambio, si se grafican todas las conexiones entre los nodos, se considera un grafo no ponderado [19].

Para el caso específico de la Figura 1.34, el grosor de cada arista refleja la recurrencia con la que se conectan cada par de nodos.

1.3.3.2. Medidas de centralidad

De igual forma que en la estadística se disponen de medidas de tendencia central como la mediana o la moda para determinar dónde se encuentran concentrados la mayor cantidad de elementos, en la teoría de grafos se cuentan con medidas de centralidad que permiten identificar los nodos más importantes dentro de la red [19].

1.3.3.2.1. Número de nodos

No se trata de una medida de centralidad como tal, aunque proporciona una idea del tamaño de la red, pues el número de nodos presentes en la red permite determinar si se trata de una red grande o pequeña. En el caso de las redes vehiculares, el número de nodos corresponde al número de vehículos presentes en dicha red.

1.3.3.2.2. Grado de centralidad

También conocida como centralidad basada en grado, es la medida más básica y corresponde al número de enlaces conectan a un nodo con los demás y permite medir así la importancia de un actor respecto a sus vecinos más cercanos, aunque, no considera la estructura global de la red [20].

Cuando se dispone de un grafo dirigido, se presentan dos variantes de esta medida de centralidad, considerando solo los enlaces de salida *(out-degree)* o los enlaces de entrada *(in-degree).*

Si se toman en cuenta los enlaces de salida, un valor alto de esta métrica significa que el nodo es *prominente,* ya que el resto de los nodos procuran tener enlaces directos con este. En cambio, si se consideran los enlaces de entrada y el grado de centralidad es alto, significa que el nodo es *influyente,* por lo que podría transferir información con el resto de la red de forma rápida [19].

Esta métrica puede estar expresada de forma normalizada, para lo cual basta con dividir el número de enlaces generados en cada uno de los n nodos entre $n-1$. Esto resulta muy útil cuando se quieren comparar redes con diferentes números de elementos.

1.3.3.2.3. Intermediación

Esta medida expresa el número de veces que un nodo actúa como puente a lo largo de la ruta de menor longitud entre dos nodos. Tomando σ_{ij} como el número de rutas de distancia mínima que une a los nodos i y j, y $\sigma_{ij}(v)$ como el número de estas rutas que pasan por el nodo v, entonces la métrica de intermediación para v se calcula en base a la siguiente ecuación:

$$Intermediación\ (v) = \sum_{i \neq v \in V} \sum_{j \neq v \in V} \frac{\sigma_{ij}(v)}{\sigma_{ij}} \tag{1.1}$$

Esta métrica únicamente toma valores entre cero y el número máximo de parejas posibles que se pueden formar en la red sin considerar el nodo para el cual se está realizando el cálculo.

Los valores del grado de centralidad y de la intermediación están muy relacionados entre sí, pues mientras mayor sea el valor de intermediación para el nodo v, mayor será la importancia de este nodo dentro de la red.

1.3.3.2.4. Distancia media

La distancia entre los nodos i y j está dada por la ruta más corta que los une. De forma matemática, la distancia puede expresarse de la siguiente forma:

$$Distancia(i,j) = min[Longitud(i,j)] \tag{1.2}$$

En un grafo existen tantas distancias como pares posibles de nodos, por lo que resulta más conveniente tomar el promedio en lugar de los valores individuales. En este caso, la medida tiene el nombre de distancia media.

1.3.3.2.5. Diámetro

El diámetro corresponde al valor máximo de excentricidad, que no es más que la distancia más larga entre todas las distancias posibles [19]. En otras palabras, es el número de saltos que se deben dar para conectar los dos nodos más alejados de la red.

Para calcular el diámetro se puede utilizar la siguiente expresión:

$$Diámetro = max[Excentricidad(i)], \quad \forall i \tag{1.3}$$

1.3.3.2.6. Densidad de aristas

Esta métrica expresa la relación entre los pares de nodos conectados en la red respecto a todos los posibles. Si n es el número total de nodos en la red y M el número de pares conectados en el grafo, entonces, la densidad de aristas se calcula la siguiente ecuación:

$$Densidad = \frac{M}{n(n-1)} \tag{1.4}$$

Para el cálculo de esta medida se puede usar la matriz de adyacencia, que no es más que una matriz cuadrada binaria cuyas filas y columnas representan cada uno de los nodos, donde el uno indica que hubo conexión entre cada par de nodos y el cero indica que no se generó conexión entre ellos.

Cuando el valor de la densidad es igual a 1 (o 100%) se puede concluir que el grafo es totalmente conexo.

1.3.3.2.7. Transitividad

Esta es una proporción de triadas transitivas (Λ) en relación al número de potenciales triadas transitivas (Δ) [12]. Una triada transitiva se da cuando un nodo i está conectado a j, j está conectado a un tercer nodo k y a su vez este está conectado al nodo i. La Ecuación (1.5) permite obtener el valor de la transitividad de una red.

$$Transitividad = \frac{\Lambda}{\Delta} \tag{1.5}$$

El valor de la transitividad siempre se encontrará dentro del rango $[0,1]$. Un valor cercano o igual a 1 indica que la red contiene todas las posibles aristas (conectividad completa).

Visto de otra forma, la transitividad puede entenderse también como la posibilidad de que dos nodos i y j conectados directamente, puedan comunicarse usando un tercer nodo k.

1.3.3.2.8. Número de comunidades

Se conoce como comunidad a un subconjunto de nodos que están estrechamente relacionados entre sí. Algunas redes están formadas por comunidades (conjuntos de nodos con un gran número de conexiones en su interior, pero escasamente conectadas a otras subredes o comunidades).

En el caso de las redes vehiculares, determinar el número de comunidades formadas en un grafo puede ser útil para inferir como se forman los grupos cuando se trabaja con protocolos de enrutamiento basados en clúster [21].

1.3.4. LENGUAJE R

R es una poderosa herramienta para la computación estadística y la creación de gráficos desarrollado por Ross Ihaka y Robert Genman y como una implementación del lenguaje S creado por los Laboratorios AT&T Bell. R tiene una naturaleza doble de programa y lenguaje de programación y entre sus principales características destacan el ser de código abierto y multiplataforma [22].

R ha tomado mucha popularidad por la gran cantidad de funciones y rutinas para análisis estadísticos y gráficos, que pueden ser visualizados en su propia ventana o exportarse en diferentes formatos. Por su parte, los resultados de los análisis estadísticos también pueden guardarse, exportarse a un archivo o ser utilizados en posteriores análisis [23].

R es un lenguaje orientado a objetos interpretado y no compilado, es decir, los comandos ingresados a través del teclado son directamente ejecutados, sin la necesidad de crear un ejecutable [22]. Se considera un lenguaje orientado a objetos porque trata a todo como un objeto, es decir, las variables, datos, resultados, etc., se alojan en la memoria de la máquina en forma de objetos con un nombre específico. Estos objetos se manipulan o modifican mediante operadores (comparativos, lógicos o aritméticos) y funciones, que también son objetos.

En la Figura 1.4 se esquematiza el funcionamiento de R y nuevamente se puede notar que la esencia de R son los objetos que son guardados en la memoria activa del computador. Solo se realiza la lectura y escritura de archivos cuando se necesita ingresar o exportar datos y resultados como gráficas o tablas.

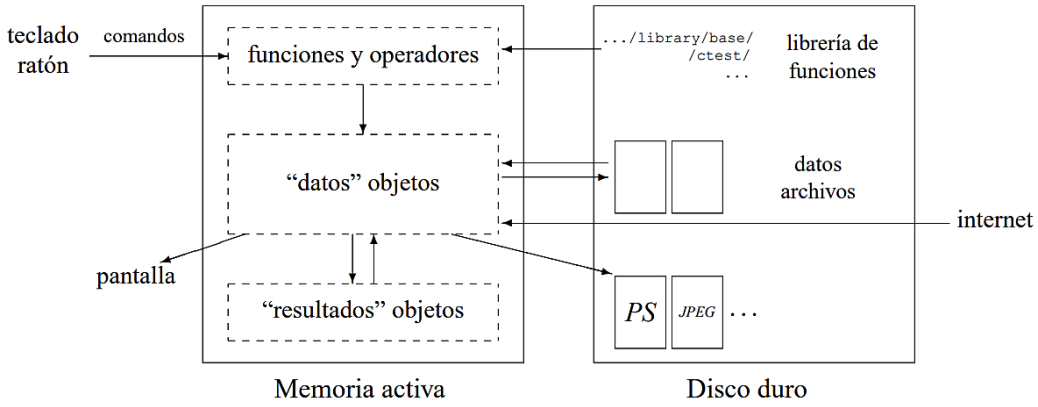

Figura 1.4. Esquema básico del funcionamiento de R. Tomado de [22]

Otra de las ventajas de R es su sintaxis simple e intuitiva, puesto que los nombres de las funciones están relacionados con su funcionamiento, por ejemplo, se puede obtener el promedio de un arreglo de números usando el comando $mean(x)$.

1.3.4.1. Librería igraph

Como se ha mencionado, existe una gran cantidad de librerías y paquetes para R que permiten aumentar sus funcionalidades y, una de ellas es la librería *igraph* que permite el análisis de redes relacionales o grafos.

El objetivo principal de esta librería es proveer un conjunto de tipos de datos, rutinas y funciones para la implementación de algoritmos de grafos, el manejo de grafos grandes con miles de nodos y para la creación rápida de prototipos a través de un lenguaje de alto nivel como R.

Para instalar esta librería basta con introducir el siguiente comando dentro de la consola de RStudio:

```
install.packages("igraph")
```

Código 1.5. Comando para la instalación de la librería igraph en R

Una vez instalado el paquete está listo para usarse y pueden calcularse las métricas descritas en la sección anterior. Para esto, únicamente se debe cargar la librería y llamar a las funciones incluidas en ella.

1.3.4.2. Cálculo de las métricas de centralidad

Para obtener el grado de centralidad se puede hacer uso de la función *degree* que tiene tres parámetros principales. El primero es el grafo sobre el cual se debe calcular la métrica. El segundo argumento *mode* permite especificar si se toman en cuenta todas las aristas,

solo las de salida o solo las de entrada. El tercer parámetro *normalized* determina si esta medida está normalizada o no (por defecto no se normaliza).

```
degree(grafo, mode = "total", normalized = TRUE)
```

Código 1.6. Función para obtener el grado de centralidad de un grafo

La intermediación se calcula usando la función *betweennes,* que recibe dos parámetros importantes. El primero es el grafo al que se le calculará la métrica y el segundo argumento es *normalized* que permite especificar si este valor estará o no normalizado. Por defecto, la función no normaliza el valor de la intermediación.

```
betweennes(grafo, normalized = TRUE)
```

Código 1.7. Función para obtener el valor de la intermediación de un grafo

La distancia promedio se puede encontrar con la función *mean_distance* que recibe un único parámetro importante que es el grafo al cual se le quiere calcular la métrica.

```
mean_distance(grafo)
```

Código 1.8. Función para calcular la distancia promedio de un grafo

El diámetro se obtiene con la función *diameter,* que recibe como parámetro el grafo al cual se le debe aplicar esta función. El resultado de función es el número de saltos que se deberían producir para conectar los dos nodos más alejados en la red.

```
diameter(grafo)
```

Código 1.9. Función para calcular el diámetro de un grafo

La densidad de aristas puede calcularse utilizando la función *edge_density,* que recibe un único parámetro importante que corresponde al grafo sobre el cual se calculará la métrica. Antes de utilizar esta función es importante que la matriz de adyacencia contenga solo dos valores (1 y 0) que representan los casos cuando exista o no conexión entre los nodos.

```
edge_density(grafo)
```

Código 1.10. Función para calcular la densidad de aristas de un grafo

De igual manera, la transitividad puede obtenerse con la función *transitivity,* que recibe como único argumento el grafo sobre el cual se debe aplicar esta función.

```
transitivity(grafo)
```

Código 1.11. Función para obtener el valor de la transitividad de un grafo

Para el caso del número de comunidades existen varias alternativas, sin embargo, uno de los métodos más conocidos es Newman-Girvanuna que se basa en la intermediación. La idea de este método es eliminar de forma secuencial las aristas que tienen el valor de intermediación más alto y recalcular esta métrica en cada paso, para seleccionar así la partición que presente una mejor segmentación de la red. Dentro de R, se puede utilizar la siguiente función para detectar las comunidades utilizando este método descrito.

```
cluster_edge_betweennes(grafo)
```

Código 1.12. Función para detectar las comunidades de un grafo

2. METODOLOGÍA

Para el desarrollo de este trabajo de titulación se efectuarán pruebas con algunas variaciones del escenario basadas en la cantidad de vehículos o densidad, período y probabilidad de reenrutamiento. El propósito es establecer las principales diferencias al cambiar cada uno de estos parámetros en términos de la conectividad que pueden llegar a tener los vehículos durante su ruta para determinar aquellos valores que permitan obtener una simulación más realista.

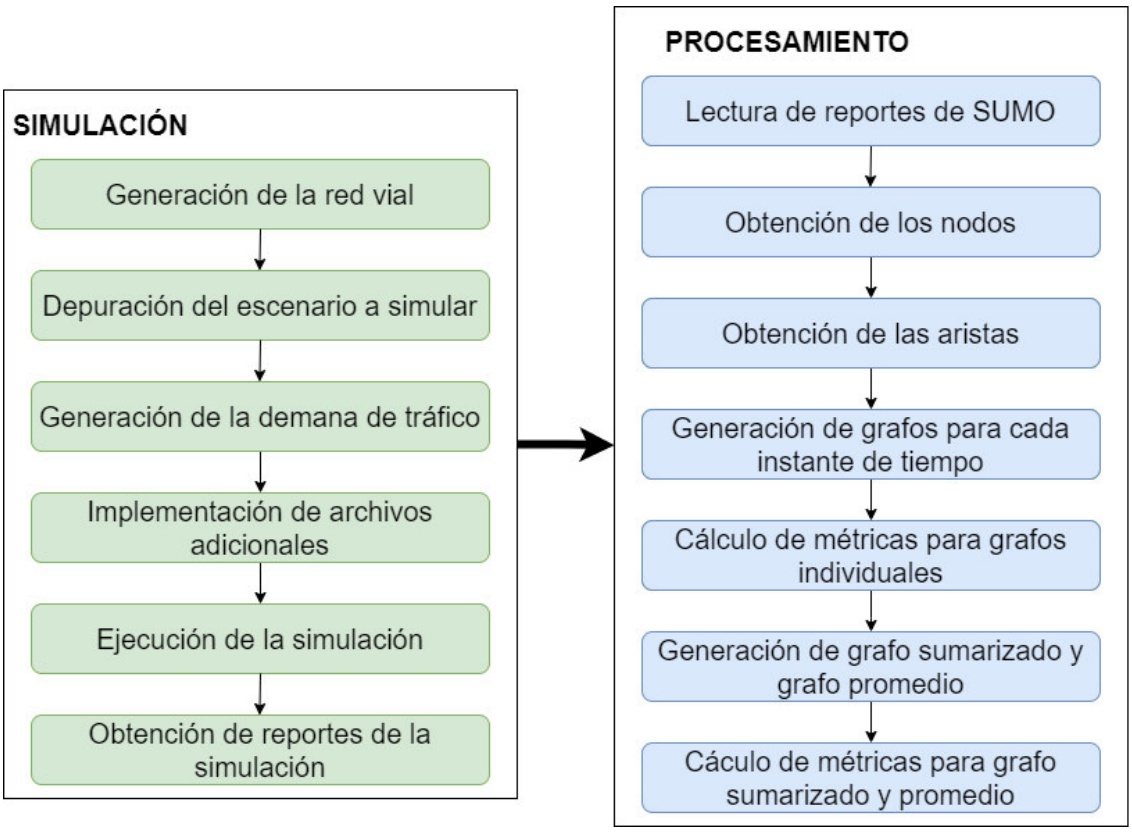

Figura 2.1. Esquema general del proceso de simulación y procesamiento

En la Figura se muestra, de forma general, los pasos seguidos para la ejecución de las simulaciones en SUMO, los scripts utilizados para el procesamiento de los archivos de salida y obtener los grafos y métricas de todos los casos hasta llegar a la obtención de las gráficas de los valores obtenidos para cada una de estas métricas.

La primera etapa consiste en la simulación del escenario, partiendo de un mapa obtenido de *Open Street Map*, la generación de la demanda de tráfico de los vehículos y la configuración del archivo principal de simulación con las distintas opciones para el reenrutamiento. De este proceso se obtienen los reportes en formato XML que sirven como entrada de la segunda etapa.

La siguiente fase es obtener la información sobre los nodos para cada intervalo de tiempo, leyendo el archivo emisión.xml que contiene las posiciones X e Y de cada vehículo. Para cada instante se escribe en un archivo diferente, el identificador de cada vehículo, su posición y tipo.

El tercer paso es obtener las aristas, es decir, determinar un valor que represente la conexión que tuvieron un par de vehículos durante la simulación. Para esto se mide el nivel de conectividad asumiendo que los vehículos estuvieron equipados con una tarjeta de red permitía formar redes entre ellos.

Una vez obtenidos los nodos y aristas se ejecuta un script de R que genera los grafos para cada uno de los instantes de tiempo y calcula los valores de las métricas

2.1. SIMULACIÓN DEL ESCENARIO SELECCIONADO

2.1.1. DEFINCIÓN DE LOS CASOS DE SIMULACIÓN

2.1.1.1. Variación en la densidad vehicular

Según el Anuario de Estadísticas de Transporte 2018 elaborado por el INEC, en ese año se contabilizaron 2’237.264 vehículos en el Ecuador, de los cuales 1’939.534 son catalogados como livianos y los restantes 297.730 como comerciales. Los automóviles y camionetas están dentro de la categoría livianos, mientras que los buses y camiones en comerciales.

En el escenario seleccionado se simularán tanto los autos livianos de uso particular como los vehículos de transporte público (buses, trolebús y ecovía). Para determinar la cantidad de vehículos que se generarán en cada caso de simulación, se utilizan las estadísticas de tráfico en las vías del Distrito Metropolitano de Quito provistas por la Secretaría de Movilidad y que se adjuntan en el Anexo A.

Los vehículos serán generados durante un período de cuatro horas, por lo que se han elegido tres variaciones de acuerdo con la cantidad de autos, de manera que se han elegido las cuatro horas con los valores más altos, más bajos y medios, que corresponden a las densidades alta, baja y media respectivamente. Para esto se han sumado los valores de todas las calles por cada hora y en función de este total se han seleccionado cada uno de los períodos.

En la Tabla 2.1 se muestra el total de vehículos a simular por cada hora seleccionada para cada uno de los tipos de densidad vehicular. Es necesario mencionar que se varían la

cantidad de vehículos particulares y se mantienen constantes los valores para los buses, trolebús y ecovía, ya que estos mantienen sus frecuencias durante todo el día.

Tabla 2.1. Número de vehículos por hora para los casos de densidad vehicular

INTERVALO	DEN. ALTA	DEN. MEDIA	DEN. BAJA
Primera hora	53217	46963	29030
Segunda hora	52419	50510	35528
Tercera hora	53808	48917	25930
Cuarta hora	53108	43490	17014
TOTAL	**212552**	**189880**	**107502**

Con los datos de la tabla anterior se define el primer parámetro que se varía en los casos de simulación, que corresponde a la cantidad de vehículos en cada hora.

2.1.1.2. Variaciones de la probabilidad de reenrutamiento

Como se mencionó en el capítulo anterior, una de las opciones que se pueden configurar como parte del reenrutamiento, es la probabilidad que tienen los vehículos de cambiar su ruta. En primera instancia, esto puede llegar a afectar significativamente la simulación ya que, con una probabilidad muy baja, pocos vehículos buscarán una nueva ruta y en un escenario con un nivel significativo de congestión, no se evidenciará una mejora con el uso de esta opción. De igual forma, con una probabilidad muy alta (cercana al 100%) todos los vehículos estarán constantemente calculando una ruta más corta para llegar a su destino, lo que puede ocasionar congestión en un escenario que en el inicio no presenta estos problemas.

Para medir los efectos de variar este parámetro, se plantean cuatro casos relacionados con la probabilidad que se presentan en la siguiente tabla:

Tabla 2.2. Valores de probabilidad para la simulación

PROBABILIDAD	DESCRIPCIÓN
0%	Ningún vehículo cambiará su ruta durante la simulación
25%	Probabilidad de reenrutamiento baja
50%	Probabilidad de reenrutamiento media
75%	Probabilidad de reenrutamiento alta

2.1.1.3. Variaciones del período de reenrutamiento

Otra de las opciones permitidas en la configuración del reenrutamiento es el intervalo que los vehículos esperan para buscar una nueva ruta más corta. Este parámetro tiene un

impacto grande en la simulación, puesto que, si este período es muy pequeño, se demandará una mayor capacidad y cantidad de recursos computacionales porque los vehículos estarán constantemente revisando el estado de la red vial para calcular una nueva ruta hacia su destino. Además, un escenario así puede llegar a ser un tanto irrealista ya que, en la aplicación real, los conductores no buscan rutas alternativas con mucha frecuencia. Para analizar los efectos de la variación de este período, se plantean cuatro valores diferentes: 1 min, 3 min, 5 min y 10 min.

Tomando en cuenta todas las combinaciones posibles se arman todos los casos posibles para la simulación y estos se presentan en la Tabla 2.3. El nombre de cada caso de simulación se forma con la densidad, período y probabilidad respectiva.

Tabla 2.3. Lista de los casos de simulación

DENSIDAD	PERÍODO	PROBABILIDAD	ESCENARIO
Baja		0.00	densidad-baja-NR
	1	0.25	densidad-baja-R1-0.25
		0.50	densidad-baja-R1-0.50
		0.75	densidad-baja-R1-0.75
	3	0.25	densidad-baja-R3-0.25
		0.50	densidad-baja-R3-0.50
		0.75	densidad-baja-R3-0.75
	5	0.25	densidad-baja-R5-0.25
		0.50	densidad-baja-R5-0.50
		0.75	densidad-baja-R5-0.75
	10	0.25	densidad-baja-R10-0.25
		0.50	densidad-baja-R10-0.50
		0.75	densidad-baja-R10-0.75
Media		0.00	densidad-media-NR
	1	0.25	densidad-media-R1-0.25
		0.50	densidad-media-R1-0.50
		0.75	densidad-media-R1-0.75
	3	0.25	densidad-media-R3-0.25
		0.50	densidad-media-R3-0.50
		0.75	densidad-media-R3-0.75
	5	0.25	densidad-media-R5-0.25
		0.50	densidad-media-R5-0.50
		0.75	densidad-media-R5-0.75
	10	0.25	densidad-media-R10-0.25
		0.50	densidad-media-R10-0.50
		0.75	densidad-media-R10-0.75
Alta		0.00	densidad-alta-NR
	1	0.25	densidad-alta-R1-0.25
		0.50	densidad-alta-R1-0.50
		0.75	densidad-alta-R1-0.75
	3	0.25	densidad-alta-R3-0.25
		0.50	densidad-alta-R3-0.50
		0.75	densidad-alta-R3-0.75
	5	0.25	densidad-alta-R5-0.25
		0.50	densidad-alta-R5-0.50
		0.75	densidad-alta-R5-0.75
	10	0.25	densidad-alta-R10-0.25
		0.50	densidad-alta-R10-0.50
		0.75	densidad-alta-R10-0.75

2.1.2. GENERACIÓN DE LA RED VIAL

2.1.2.1. Importación del mapa desde Open Street Map

Una vez que se han definido todos los casos para la simulación, es necesario el escenario en el cual se probarán todos estos casos. Para este trabajo de titulación se utilizarán los mapas caracterizados y validados en [21]. El área de estudio seleccionada tiene 115 km^2 se presenta en la Figura 2.2 y corresponde a los alrededores del parque La Carolina y el mapa de esta zona se obtiene directamente de la página web de *Open Street Map*.

Figura 2.2. Zona de estudio vista en OSM

Por las dimensiones del área seleccionada, se debe descargar por partes y usando la herramienta *Netconverter* [24] se unen estos archivos en un solo mapa y se convierte a una red vial aceptada por SUMO. Las principales configuraciones que se realizan en la herramienta es la opción *no-turnarounds* para deshabilitar la creación automática de giros en las intersecciones de las vías, lo que previene la creación de cambios de vía ficticios e imprudentes, así como giros que en la realidad no están permitidos en algunas intersecciones. También se utiliza la opción *geometry.remove* para asegurar la continuidad de la calle a pesar de los cambios en su geometría, caso contrario se creará una nueva vía en el momento que haya un cambio en la geometría. El efecto de esta opción se puede

observar en la Figura 2.2. Cuando se trabaja con el valor por defecto de esta opción, se crean de forma innecesaria nuevas calles.

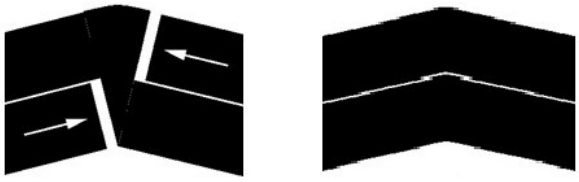

Figura 2.3. Efecto de usar la opción *geometry.remove* en Netconverter

Adicionalmente, se utilizan las opciones *remove.edges.isolated* para eliminar vías desconectadas con el fin de evitar simular vehículos allí y *tls.join* para combinar todos los semáforos de un cruce bajo el mismo identificador para facilitar su gestión.

2.1.2.2. Depuración del escenario

Una vez que se ha finalizado el proceso de importación del mapa, es necesario depurar de forma manual los posibles errores que pudieron existir con la herramienta para importar. Para esto se hace uso de otra herramienta de SUMO llamada *Netedit* [25] que toma como entrada el archivo net.xml generado en el paso anterior. En esta etapa principalmente se corrigen errores asociados con los giros permitidos en las intersecciones, se eliminan las vías exclusivas sobrantes, ya que desde OMS pueden importarse vías exclusivas para peatones, bicicletas, motos, que en la realidad no están presentes.

En las Figuras 2.4 y 2.5 se muestran algunos ejemplos de los errores más comunes que fueron corregidos durante esta etapa. La primera muestra, como se controlan los giros en los cruces que disponen de semáforos y con este proceso se controlan los cambios no permitidos.

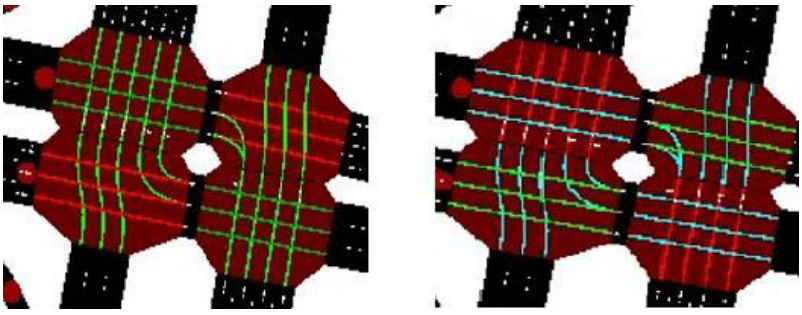

Figura 2.4. Depuración de errores en intersección con semáforo. (izquierda) Default, (derecha) Corregido

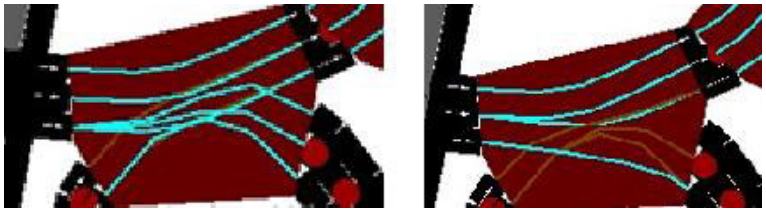

Figura 2.5. Depuración de errores en redondel. (izquierda) Default, (derecha) Corregido

En la figura anterior, se puede observar los errores en redondeles, donde no todos los giros están permitidos, y mediante la edición manual de la red vial, se corrigen estos problemas para disponer de un escenario de simulación lo más realista posible.

Una vez que se termina el proceso de depuración y afinamiento de la red vial, se dispone del mapa listo para ser utilizado en la simulación con SUMO y en la figura 2.6 se visualiza este mapa final usando la herramienta *netedit.*

Figura 2.6. Mapa resultante del proceso de importación y depuración

2.1.3. GENERACIÓN DE LA DEMANDA DE TRÁFICO

2.1.3.1. Generación de demanda para buses y articulados

Se define primero la demanda de tráfico para los buses, trolebús y ecovía ya que, como se mencionó al inicio de este capítulo, estos serán constantes en todos los casos de simulación planteados porque sus frecuencias son las mismas sin importar la cantidad de vehículos particulares que circulen por las vías.

El primer paso es generar las paradas para cada una de las líneas de buses y estos datos se obtienen de la página web de la Secretaría de Movilidad de Quito [26]. En el código 2.1 se muestra un extracto del archivo *busStop.xml* que contiene las paradas para los buses, trolebús y ecovía. Considerando la normativa de Tránsito, estos vehículos deben circular únicamente por el carril derecho en aquellas vías en las que se dispongan de dos o más carriles, por lo que dentro de cada etiqueta *busStop* se configura el identificador de la calle junto con el identificador del carril que en todos los casos es el número 0.

```
<a>
  <!--Paradas del Trole-->
  <busStop id="troleA1" lane="-423658293_0" startPos="210" endPos="230"
name="Colón_SN"/>
  <busStop id="troleB1" lane="309391191#1_0" startPos="850" endPos="870"
name="Colón_NS"/>
  <!-- Código omitido -->

  <!--Paradas de la Ecovía-->
  <busStop id="ecoviaA1" lane="150172867_0" startPos="60" endPos="80"
name="Orellana_SN"/>
  <busStop id="ecoviaB1" lane="150172873_0" startPos="70" endPos="90"
name="Orelana_NS"/>
  <busStop id="ecoviaA2" lane="150172872_0" startPos="50" endPos="70"
name="LaPaz_SN"/>
  <!-- Código omitido -->

  <!--Paradas de Buses-->
  <busStop id="stop1" lane="424798280_0" startPos="45" endPos="70" name="Orellana-
Coruña"/>
  <busStop id="stop24" lane="532145753#2_0" startPos="220" endPos="245" name="Shyris-
NNUU"/>
  <busStop id="stop25" lane="537702926#1_0" startPos="40" endPos="65" name="Shyris-
NNUU"/>
  <busStop id="stop45" lane="432815431#0_0" startPos="110" endPos="135"
name="Carolina"/>
  <busStop id="stop46" lane="432815431#3_0" startPos="60" endPos="85" name="CCI"/>
  <!-- Código omitido -->
</a>
```

Código 2.1. Archivo de paradas de buses, trolebús y ecovía

Para el área elegida se disponen de 18 líneas de buses autorizadas que tienen rutas tanto de ida como de regreso, por lo que se deben generar 36 flujos vehiculares para los flujos. Como las rutas se mantienen fijas, resulta más sencillo especificar la ruta completa para cada flujo, así como las paradas que debe realizar durante su recorrido. En el siguiente código se muestra una parte del archivo *buses.rou.xml* en el que se indica entre otras cosas el tipo de vehículo, el factor de velocidad, los tiempos de inicio y finalización del flujo, el intervalo con el cual será insertado cada bus.

```
<flows>
    <vType id="bus" vClass="bus" color="blue" speedFactor="normc(0.8,0.07,0.5,1.1)"/>
    <flow id="Roldos-Jardin" from="531413901" to="24406394" begin="0" end="14400"
period="420" type="bus" line="AguilaDorada">
        <route edges="531413901 537702923 531415382#0 531415382#1 531415382#2
537702925 531415381 537675272 531415931 553686773 531415930 431945826#0 431945826#1
537702926#0 537702926#1 531418095 291018416 399552669 531420097#0 531420097#1
531420097#2 531420520 531420519 540374181 531421000#0 531421000#1 531424822
431777058#1 531424823#0 531424823#1 531424820 -89636470 24406394"/>
        <stop busStop="stop32" duration="10"/>
        <stop busStop="stop30" duration="10"/>
        <stop busStop="stop28" duration="10"/>
        <stop busStop="stop25" duration="10"/>
        <stop busStop="stop23" duration="10"/>
        <stop busStop="stop22" duration="10"/>
        <stop busStop="stop20" duration="10"/>
    </flow>
    <!-- Código omitido -->
<flows>
```

Código 2.2. Demanda de tráfico para líneas de buses

El siguiente paso es generar la demanda de tráfico para los articulados (trolebús y ecovía), no obstante, para este caso se utiliza la herramienta *DUAROUTER* para generar las rutas de estos vehículos a partir de un archivo de flujos en el que se indican únicamente los puntos de partida y destino, como se muestra en el Código 2.3

```
<flows>
    <vType id="trolebus" vClass="bus" guiShape="bus/flexible" length="18" width="2.5"
height="3" speedFactor="normc(0.80,0.02,0.70,0.90)"/>

    <flow id="a" from="150172867" to="150172870#6" begin="0" end="14400" period="120"
type="trolebus" via="-150172877#2 -31924758#3 -150172876#1 -150172879#2 -150172879#0
150172862#2 -150172874#1 150172870#2" color="red">
        <stop busStop="ecoviaA1" duration="15"/>
        <stop busStop="ecoviaA2" duration="15"/>
        <stop busStop="ecoviaA3" duration="15"/>
        <stop busStop="ecoviaA4" duration="15"/>
        <stop busStop="ecoviaA5" duration="15"/>
        <stop busStop="ecoviaA6" duration="15"/>
        <stop busStop="ecoviaA7" duration="15"/>
        <stop busStop="ecoviaA8" duration="15"/>
```

```
        <stop busStop="ecoviaA9" duration="15"/>
        <stop busStop="ecoviaA0" duration="15"/>
    </flow>

    <!-- Código omitido -->

    <flow id="c" from="-423658293" to="-423441354#0" begin="0" end="14400"
period="120" type="trolebus" color="green">
        <stop busStop="troleA1" duration="15"/>
        <stop busStop="troleA2" duration="15"/>
        <stop busStop="troleA3" duration="15"/>
        <stop busStop="troleA4" duration="15"/>
        <stop busStop="troleA5" duration="15"/>
        <stop busStop="troleA6" duration="15"/>
        <stop busStop="troleA7" duration="15"/>
    </flow>

    <!-- Código omitido -->

</flows>
```

Código 2.3. Archivo de flujos para trolebús y ecovía

Ahora se crea un archivo de configuración para *DUAROUTER* en el que se indican los archivos de entrada: flujos de trolebús y ecovía, red vial y paradas y el archivo que será generado a la salida y contiene las rutas para cada vehículo generado. En los Códigos 2.4 y 2.5 se muestra el archivo de configuración y una porción de las rutas generadas con la herramienta, respectivamente.

```
<configuration>
    <input>
        <net-file value="mapa.net.xml"/>
        <route-files value="trolebus.xml"/>
        <additional-files value="busStop.xml"/>
    </input>
    <output>
        <output-file value="trolebus.rou.xml"/>
    </output>
</configuration>
```

Código 2.4. Archivo de configuración de DUAROUTER para trolebús y ecovía

```
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/routes_file.xsd">
    <vType id="trolebus" length="18.00" speedFactor="normc(0.80,0.02,0.70,0.90)"
vClass="bus" guiShape="bus/flexible" width="2.50" height="3.00"/>
    <vehicle id="a.0" type="trolebus" depart="0.00" color="red">
        <routeDistribution last="0">
            <route cost="380.60" probability="1.00000000" edges="150172867 -
150172877#4 -150172877#3 -150172877#2 -150172877#1 -150172877#0 150172872 -31924758#3
-31924758#2 -31924758#1 -31924758#0 150172868 150172868.23 -150172876#1 -150172876#0
150172858 -150172879#3 -150172879#2 -150172879#1 -150172879#0 150172863 150172862#1
150172862#2 -150172874#2 -150172874#1 -150172874#0 150172870#0 150172870#1
150172870#2 532919578#2 150172870#4 150172870#5 150172870#6"/>
```

```
        </routeDistribution>
        <stop busStop="ecoviaA1" duration="15.00"/>
        <stop busStop="ecoviaA2" duration="15.00"/>
        <stop busStop="ecoviaA3" duration="15.00"/>
        <stop busStop="ecoviaA4" duration="15.00"/>
        <stop busStop="ecoviaA5" duration="15.00"/>
        <stop busStop="ecoviaA6" duration="15.00"/>
        <stop busStop="ecoviaA7" duration="15.00"/>
        <stop busStop="ecoviaA8" duration="15.00"/>
        <stop busStop="ecoviaA9" duration="15.00"/>
        <stop busStop="ecoviaA0" duration="15.00"/>
    </vehicle>

    <!-- Código omitido -->

</routes>
```

Código 2.5. Ejemplo de las rutas generadas para trolebús y ecovía

2.1.3.2. Generación de la demanda de tráfico para autos particulares

Para el caso de los vehículos particulares, se utiliza también la herramienta *DUAROUTER*, pero esta vez en los flujos, solo se especificará el origen desde el cual parten los autos. Utilizando los valores de cada hora seleccionada para cada densidad de tráfico, se crean los archivos *flows.xml* con una estructura similar a la que se presenta en el Código 2.6

```
<flows>
<!-- Caso: sin reenrutamiento -->
<!-- Densidad: alta -->
<!-- Total vehiculos: 102506 -->
    <vType id="auto" vClass="passenger" speedFactor="normc(1,0.07,0.6,1.3)"/>
    <interval begin="0" end="3600">
        <flow id="a00" from="532137775#0" number="1009" type="auto" departLane="free"/>
        <flow id="a01" from="424798280" number="1414" type="auto" departLane="free"/>
        <flow id="a02" from="415999035#0" number="302" type="auto" departLane="free"/>
        <flow id="a03" from="421616332#1" number="602" type="auto" departLane="free"/>
        <flow id="a04" from="-63521323#0" number="174" type="auto" departLane="free"/>
        <flow id="a05" from="24361373#2" number="650" type="auto" departLane="free"/>
        <flow id="a06" from="421924777#1" number="1516" type="auto" departLane="free"/>
        <flow id="a07" from="532919568#0" number="1134" type="auto" departLane="free"/>
        <flow id="a08" from="-31238395#1" number="211" type="auto" departLane="free"/>
        <flow id="a09" from="537626454" number="1604" type="auto" departLane="free"/>
        <flow id="a10" from="423253092#0" number="941" type="auto" departLane="free"/>
        <flow id="a11" from="531413901" number="1397" type="auto" departLane="free"/>
        <flow id="a12" from="267656596" number="1503" type="auto" departLane="free"/>
        <flow id="a13" from="31820582#0" number="825" type="auto" departLane="free"/>
        <flow id="a14" from="150178874" number="1253" type="auto" departLane="free"/>
        <flow id="a15" from="532895209#2" number="1805" type="auto" departLane="free"/>
        <flow id="a16" from="350991849#1" number="818" type="auto" departLane="free"/>
        <flow id="a17" from="532942723#0" number="719" type="auto" departLane="free"/>
        <flow id="a18" from="24392969#1" number="1174" type="auto" departLane="free"/>
```

```
        <flow id="a19" from="31246527#2" number="1134" type="auto" departLane="free"/>
        <flow id="a20" from="176088062#0" number="874" type="auto" departLane="free"/>
        <flow id="a21" from="533107317#1" number="1123" type="auto" departLane="free"/>
        <flow id="a22" from="-420357726#1" number="192" type="auto" departLane="free"/>
        <flow id="a23" from="429766161" number="2389" type="auto" departLane="free"/>
        <flow id="a24" from="-420360412" number="896" type="auto" departLane="free"/>
        <flow id="a25" from="-415957380#1" number="373" type="auto" departLane="free"/>
        <flow id="a26" from="-538965913#1" number="291" type="auto" departLane="free"/>
        <flow id="a27" from="34396683#3" number="525" type="auto" departLane="free"/>
        <flow id="a28" from="420385365#0" number="785" type="auto" departLane="free"/>
    </interval>
<interval begin="3600" end="7200">
        <flow id="b00" from="532137775#0" number="1031" type="auto" departLane="free"/>
        <flow id="b01" from="424798280" number="1474" type="auto" departLane="free"/>
        <flow id="b02" from="415999035#0" number="337" type="auto" departLane="free"/>
        <flow id="b03" from="421616332#1" number="596" type="auto" departLane="free"/>
        <flow id="b04" from="-63521323#0" number="240" type="auto" departLane="free"/>
        <flow id="b05" from="24361373#2" number="669" type="auto" departLane="free"/>
        <flow id="b06" from="421924777#1" number="1426" type="auto" departLane="free"/>
        <flow id="b07" from="532919568#0" number="1181" type="auto" departLane="free"/>
        <flow id="b08" from="-31238395#1" number="216" type="auto" departLane="free"/>
        <flow id="b09" from="537626454" number="1389" type="auto" departLane="free"/>
        <flow id="b10" from="423253092#0" number="992" type="auto" departLane="free"/>
        <flow id="b11" from="531413901" number="1090" type="auto" departLane="free"/>
        <flow id="b12" from="267656596" number="1451" type="auto" departLane="free"/>
        <flow id="b13" from="31820582#0" number="770" type="auto" departLane="free"/>
        <flow id="b14" from="150178874" number="1076" type="auto" departLane="free"/>
        <flow id="b15" from="532895209#2" number="1424" type="auto" departLane="free"/>
        <flow id="b16" from="350991849#1" number="741" type="auto" departLane="free"/>
        <flow id="b17" from="532942723#0" number="809" type="auto" departLane="free"/>
        <flow id="b18" from="24392969#1" number="752" type="auto" departLane="free"/>
        <flow id="b19" from="31246527#2" number="1161" type="auto" departLane="free"/>
        <flow id="b20" from="176088062#0" number="958" type="auto" departLane="free"/>
        <flow id="b21" from="533107317#1" number="926" type="auto" departLane="free"/>
        <flow id="b22" from="-420357726#1" number="187" type="auto" departLane="free"/>
        <flow id="b23" from="429766161" number="1398" type="auto" departLane="free"/>
        <flow id="b24" from="-420360412" number="828" type="auto" departLane="free"/>
        <flow id="b25" from="-415957380#1" number="448" type="auto" departLane="free"/>
        <flow id="b26" from="-538965913#1" number="212" type="auto" departLane="free"/>
        <flow id="b27" from="34396683#3" number="497" type="auto" departLane="free"/>
        <flow id="b28" from="420385365#0" number="840" type="auto" departLane="free"/>
    </interval>
    <!-- Código omitido -->
</flows>
```

Código 2.6. Ejemplo de las rutas generadas para trolebús y ecovía

Cada flujo tiene un identificador diferente, el identificador de la calle de origen, el número de vehículos que serán insertados y se especifica el tipo de vehículo como *auto,* que indica a SUMO que se trata de un vehículo de tipo particular.

Para cada caso de densidad (alta, media y baja) se disponen de cuatro intervalos que corresponden a las cuatro horas seleccionadas para cada período. En todos los archivos,

los identificadores de los flujos y de las vías se mantienen fijos y únicamente cambian la cantidad de autos.

Ahora, se genera el archivo de configuración de *DUAROUTER* para generar las rutas para cada uno de los vehículos. Al igual que en la configuración para el trolebús y ecovía, se precisan los archivos de entrada y salida, pero se configuran adicionalmente las opciones de procesamiento tal como se muestran en el siguiente fragmento de código:

```
<configuration>
    <input>
        <net-file value="mapa.net.xml"/>
        <route-files value="flows.xml"/>
        <additional-files value="busStop.xml"/>
    </input>
    <output>
        <output-file value="autos.rou.xml"/>
    </output>
    <processing>
        <routing-algorithm value="dijkstra"/>
        <randomize-flows value="true"/>
        <ignore-errors value="true"/>
    </processing>
</configuration>
```

Código 2.7. Archivo de configuración de DUAROUTER para autos particulares

Se utiliza la opción *routing-algorithm* para especificar el algoritmo Dijkstra que toma en cuenta la ruta más corta. Además, se utilizan las opciones *randomize-flows* para generar tiempos de salida aleatorios para los flujos e *ignore-errors* para continuar con la ejecución, aunque una ruta no haya podido construirse correctamente.

2.1.4. GENERACIÓN DE ARCHIVOS ADICIONALES

Como último paso antes de ejecutar la simulación, se genera un archivo de configuración principal en el que se indican los todos los archivos que utilizará SUMO como entrada. Por un lado, se tienen los archivos de la red y los archivos que tienen las rutas para todos los vehículos que componen la simulación (autos particulares, buses, trolebús y ecovía) y se incluyen como archivos adicionales las paradas y el archivo *mapa.poly.xml* que contiene los obstáculos como edificaciones, relieve, etc.

Además, se configuran tres reportes que serán generados como salida de la simulación. El primero es *tripInfo,* que contiene principalmente información sobre la hora de salida y llegada de cada vehículo. El segudo es *emissionOutput* que presenta información acerca de las emisiones de todos los autos dentro de la red, así la posición de cada uno para cada instante de tiempo. El último reporte generado es *summary* que muestra el número de

vehículos que se encuentran cargados, en funcionamiento, esperando y aquellos que han llegado a su destino y el tiempo que necesitaron para completar su ruta.

Se usa también la opción de procesamiento *ignore-route-errors* para asegurar que la simulación siga ejecutándose a pesar de que se presenten errores con alguna ruta. Si esta opción no se configura, la simulación termina en el momento que ocurre el primer error.

En este archivo también se deben configurar todas las opciones para el reenrutamiento. Los valores de probabilidad e intervalo de reenrutamiento varían de acuerdo con los casos de simulación, no obstante, se configuran también los parámetros *rerouting-pre-period* que tiene un valor fijo de 300 segundos y corresponde al intervalo de tiempo que se espera antes de empezar con la búsqueda de una nueva ruta. La opción *rerouting-determinstic* en falso permite indicar a SUMO que debe utilizar un modelo determinístico en la simulación, ya que no se tiene certeza de que la salida será siempre la misma con estos datos de entrada. Por último, se configura la opción *rerouting-explicit* como *auto* para especificar que únicamente participarán del reenrutamiento los autos particulares, pues no tiene sentido que el resto vehículos de transporte público cambien sus rutas.

```
<configuration>
    <input>
        <net-file value="mapa.net.xml"/>
        <route-files value="autos.rou.xml,buses.rou.xml,trolebus.rou.xml"/>
        <additional-files value="mapa.poly.xml,busStop.xml"/>
    </input>
    <output>
        <tripinfo-output value="reportes/tripinfo.xml"/>
        <emission-output value="reportes/emission.xml"/>
        <summary value="reportes/summary.xml"/>
    </output>
    <processing>
        <ignore-route-errors value="true"/>
    </processing>
    <routing>
        <device.rerouting.probability value="0.25"/>
        <device.rerouting.period value="60"/>
        <device.rerouting.pre-period value="300"/>
        <device.rerouting.deterministic value="false"/>
        <device.rerouting.explicit value="auto"/>
    </routing>
    <time>
        <step-length value="1"/>
    </time>
</configuration>
```

Código 2.8. Archivo de configuración principal de la simulación en SUMO

En el código anterior se presenta la estructura completa del archivo de configuración con todas las consideraciones que se han mencionado; sin embargo, para los casos propuestos

en los cuales la probabilidad de reenrutamiento es cero, se omite la sección *routing,* puesto que es equivalente a mantener el código y asignar este valor en el parámetro correspondiente.

Para todos los casos de simulación, se generan los archivos de demanda de los vehículos y el archivo de configuración con los valores correspondientes. A modo de resumen, en la Figura 2.7 se muestran los archivos necesarios para cada simulación.

Figura 2.7. Archivos necesarios para la simulación de cada escenario

Se ejecutan las simulaciones pasando como única entrada el archivo *simular.sumo.cfg* y al final de cada ejecución se dispondrá de un directorio denominado *reportes* en el que se pueden encontrar los ficheros con la información de la simulación.

2.2. PROCESAMIENTO DE LOS ARCHIVOS DE SALIDA

Conociendo que los reportes generados por SUMO se encuentran en formato XML y pueden llegar a tener millones de líneas de código, es necesario utilizar una herramienta que permita procesar esta información de una manera eficiente y simple. Para este fin se utiliza AWK, que con un código bastante simple permite leer estos ficheros y manipular los datos dentro de ellos.

En el capítulo anterior, se mencionó que los elementos principales en un grafo son los nodos y las aristas, y aplicando este concepto al tráfico, cada vehículo dentro de la red corresponde a un nodo en específico, mientras que las aristas están representadas por la conectividad de este vehículo con los demás.

Entonces, tomando la posición de cada vehículo en cada instante de tiempo, se puede obtener los nodos con una representación de cómo están distribuidos en un plano XY. Con estos nodos, se puede obtener un valor que represente la conectividad que llega a tener con el resto de los autos que lo rodean.

Toda la lógica para el procesamiento de los datos se divide en varios archivos para evitar un solo archivo con muchas líneas de código, y la llamada a estos programas se realiza desde un script principal.

2.2.1. OBTENCIÓN DE DATOS

2.2.1.1. Obtención de los nodos

El reporte de SUMO que contiene más información acerca de la posición de los vehículos en cada instante de tiempo es el archivo *emisión.xml* ya que presenta tanto los valores de las coordenadas los planos X e Y. La estructura general de este archivo consiste en una etiqueta *timestep* que indica corresponde a un instante de tiempo y dentro de este se encuentran varios elementos identificados con la etiqueta *vehicle* que representan la información de cada vehículo. En el Código 2.9 se muestra un fragmento del archivo de emisiones para evidenciar la estructura descrita.

```
<emission-export>
    <timestep time="0.00">
        <vehicle id="Batán-Colmena.0" eclass="HBEFA3/Bus" CO2="5286.11" CO="20.17"
        HC="4.85"       NOx="60.75" PMx="2.01" fuel="2.25" electricity="0.00"
        noise="67.11" route="!Batán-Colmena" type="bus" waiting="0.00"
        lane="532919568#0_0" pos="12.10" speed="0.00" angle="256.99" x="3607.01"
        y="4410.67"/>
    <!-- Código omitido -->
</emission-export>
```

Código 2.9. Fragmento del reporte emisión.xml

Para desarrollar el programa en AWK y obtener la posición de los vehículos en cada instante de tiempo se sigue una estructura similar a la presentada en el Código 1.4. Dentro de la sentencia *begin* se define las comillas dobles (“) como el separador de campos *(FS)* y se leen los tiempos de inicio, fin y salto como argumentos de entrada, se toman los argumentos desde el número 2 porque el primero corresponde al reporte de SUMO que debe ser procesado. Se imprimen en pantalla los valores recibidos y se declaran un variable *contador* y un arreglo *nombres* que servirán para almacenar los identificadores de los vehículos en orden de aparición dentro del archivo; se inicializa el *contador* en 1 y se asigna un valor al primer elemento del arreglo.

```
FS = "\""
nodesOut="nodes"#Archivo de salida
step=ARGV[2]
ini=ARGV[3]
fin=ARGV[4]
printf ("step %s\n", step)
printf ("ini %s\n", ini)
printf ("fin %s\n", fin)
contador=1;
nombre["sinNombre"]=0
```

Código 2.10. Inicialización de variables y lectura de argumentos de entrada en AWK

Ahora se deben recorrer cada una de las líneas del archivo y tomando en cuenta el separador elegido, para las etiquetas *timestep* se puede notar que estas tienen únicamente dos columnas. La segunda columna contiene el tiempo actual y es la que se almacena en una variable *time;* se valida que este valor se encuentre dentro del rango y que sea un múltiplo del salto de tiempo especificado. Si se cumplen todas estas condiciones, la variable *nodesOut* se actualiza con el valor de *time.*

Las filas que tienen información sobre los vehículos son aquellas que tienen 6 o más columnas, entonces, se valida aquellas filas que cumplen esta condición y corresponden al instante de tiempo de interés. De cada una se obtiene el tipo de vehículo y su posición, además se hace una validación adicional para almacenar el identificador del vehículo en el arreglo *nombres.* Sin embargo, los tipos de vehículos generados en el reporte resultan poco descriptivos puesto que son cadenas de texto poco legibles, por lo que se mapean estos valores a 1 o 2 dependiendo si se trata de autos particulares o buses, respectivamente. Por último, se imprimen los datos recolectados del vehículo en el archivo de texto correspondiente usando la función *printf.*

```
if(NF==3 && $1="<timestep time"){
    time=int($2)
    if(time>=ini && time<=fin &&(time%step)==0){
        nodesOut="data\\nodes"time".txt"
        printf ("Creando archivo %s\n", nodesOut)
    }
}
if(NF>6 && time>=ini && time<=fin && (time%step)==0)
{
    id=$2
    if (!nombre[id]){
        nombre[id]=contador;
        contador++;
```

```
    }
    posx=$36
    posy=$38
    type=$4

    if(type=="HBEFA3/Bus")
        Type=2
    if(type=="HBEFA3/PC_G_EU4")
        Type=1
    printf("%i %f %f %i\n",nombre[id], posx, posy,Type)>>nodesOut;
}
if(NF>6&&time>fin)
    exit
}
```

Código 2.11. Obtención de la posición y tipo de vehículos para cada instante de tiempo

Como salida de este programa se generan tantos archivos de texto con la información de vehículos como pasos haya dentro del rango de tiempo indicado en la entrada. En el Código 2.12 se muestra un fragmento de uno de los archivos generados.

```
Id PosX PosY Type
1 1388.03 1525.86 2
2 1909.98 4467.13 2
3 1650.91 829.37 2
4 2041.26 2275 2
5 2835.7 2626.2 2
6 2101.61 1608.76 2
7 2517.75 4117.86 2
8 2460.51 4499.01 2
9 2024.92 2029.05 2
10 1564.08 2425.03 2
11 2409.87 2119.13 2
12 1217.9 898.84 2
13 1489.77 2112 2
14 1531.47 4600.93 2
15 1161.42 2469.29 2
16 2135.67 3219.11 2
17 803.65 3187.59 2
18 2031.66 2189.8 2
19 833.85 3459.03 2
20 2288.41 3524.36 2
21 1145.03 964.27 2
22 1664.94 825.63 2
23 2111.35 1631.37 2
24 2260.55 3488.45 2
25 830.06 3489.07 2
# Código omitido
```

Código 2.12. Archivo de nodos generado con el programa *getNodes.awk*

2.2.1.2. Obtención de las aristas

Una vez que se disponen de los nodos, se utiliza el programa *getEdges.exe* desarrollado en [12] para obtener las aristas de los grafos. Para esto el programa define algunos parámetros de la red vehicular que se resumen en la Tabla 2.4.

Tabla 2.4. Archivo de nodos generado con el programa de AWK

PARÁMETRO	VALOR
Potencia de transmisión	23 dBm
Umbral de sensibilidad	-82 dBm
Rango de transmisión	400 metros con línea de vista (LOS)
Modelo de atenuación	IEEE 802.11p empírico

Este programa asume que todos los vehículos en la simulación están equipados con una tarjeta de red compatible con el estándar IEEE 802.11p y tiene definida una potencia de 23

dBm, con lo que el posible rango de comunicación es de aproximadamente 400 metros a la redonda. Considera el modelo de atenuación del edificio propuesto en [27] para modelar una condición sin línea de visión (NLOS) y generar la información de las edificaciones presentes en el escenario.

Tomando en cuenta el archivo de entrada con los nodos, es decir las posiciones de los vehículos en ese instante de tiempo, la potencia de transmisión del dispositivo de cada uno, el rango aproximado de transmisión y el modelo de atenuación, se calcula la potencia recibida en el resto de los nodos y solamente cuando esta es mayor al umbral de recepción de -82 dBm se imprimen los datos de la conexión: el nodo origen, el nodo destino y la potencia en el receptor.

A la salida de este programa se genera un archivo de texto que tiene todas las conexiones entre los nodos, como se muestra en el fragmento del Código 2.13. Se puede evidenciar que los nodos logran conectarse con más de un vehículo diferente a la vez, aunque varían los valores de la potencia recibida por cada uno de estos.

```
orig dest strengh
1 34 -77
2 123 -56
2 124 -68
2 125 -72
3 22 -50
3 78 -76
3 79 -76
3 80 -76
3 91 -82
4 18 -76
4 56 -56
4 61 -60
6 23 -55
6 39 -64
8 62 -65
8 121 -67
8 122 -69
9 18 -73
9 45 -61
10 49 -77
12 46 -49
13 63 -52
14 129 -71
```

Código 2.12. Archivo de aristas generado con el programa *getEdges.exe*

2.2.1.3. Obtención de datos para el rango de tiempo

Para utilizar los dos programas descritos en las secciones anteriores para la obtención de los nodos y las aristas, se desarrolla en lenguaje R un script que permite automatizar esta tarea. Para esto el programa recibe como argumentos de entrada el directorio de trabajo, el reporte de emisiones de SUMO, el archivo de obstáculos, el tiempo de inicio, el tiempo de fin y el salto de tiempo.

Para calcular los tiempos de inicio y fin, se considera el funcionamiento de SUMO que inserta los vehículos dentro de la red vial cada cierto tiempo, por lo que no tiene sentido incluir los primeros instantes de la simulación en los que hay pocos autos o los últimos instantes en los que el comportamiento es similar, con pocos vehículos que están terminando sus viajes.

Con base en los tiempos de simulación de los casos propuestos que se presentan en el Anexo B, el promedio es 15.200 segundos y tomando en cuenta la situación anterior, se determina el rango de información útil entre el 25% y 75% de este valor. En otras palabras, el primer instante que se toma en cuenta para el procesamiento es cuando *timestep* es igual a 3800 y el último cuando *timestep* es 11400.

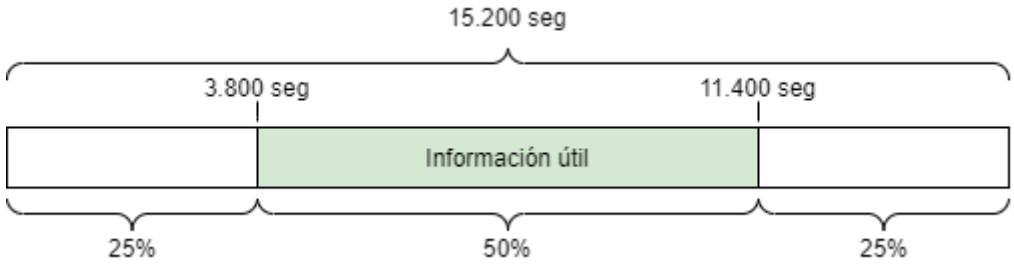

Figura 2.8. Porcentaje de información válida en los reportes de SUMO

En este programa de R, luego de la lectura de los argumentos de entrada, usando el comando *system* se ejecuta primero el programa *getNodes.exe* para generar los archivos de nodos para todos los instantes de tiempo. Luego, se iteran estos mismos tiempos mediante un lazo *for* y se ejecuta el programa *getEdges.exe* para calcular la conectividad entre estos nodos. El código completo de este programa se encuentra en el Anexo C.

2.2.2. GENERACIÓN DE GRAFOS

La conectividad de los nodos calculada en la sección anterior puede ser representada por un grafo G donde $V(G)$ es el conjunto de vértices (nodos) y $E(G)$ corresponde a las aristas. La red está representada por la matriz de adyacencia $(A_{i,j})$, donde cada entrada

corresponde al peso del borde ($E_{w_{i,j}}$), que no es más que la fuerza de la señal del enlace entre dos nodos.

$$A_{i,j} = \begin{cases} E_{w_{i,j}} & cuando\ los\ nodos\ i,j\ están\ conectados \\ 0 & cualquier\ otro\ caso \end{cases} \quad (2.1)$$

2.2.2.1. Obtención de datos individuales

Para organizar de mejor manera el código se crea un script de R llamado *functions.R* que contiene la función *get_unit_info*, para generar un grafo a partir del archivo de nodos y aristas recibidos como entrada, dibujar este grafo almacenarlo como una imagen en formato PNG *(Portable Network Graphics)*. Calcula también los valores de las métricas sobre este grafo y guarda estos valores en un arreglo. Esta función devuelve una lista con dos elementos: el grafo generado y el arreglo con los valores de las métricas.

De manera más específica, los parámetros de entrada y salida de la función se indican en la Tabla 2.5.

Tabla 2.5. Parámetros de entrada y salida de la función get_unit_info

PARÁMETRO	DESCRIPCIÓN	TIPO	E/S
Folder	Nombre de la carpeta de trabajo. Corresponde al nombre del caso simulado	String	Entrada
Time	Instante de tiempo sobre el cual se está trabajando	Integer	Entrada
nodesFile	Nombre del archivo de nodos	String	Entrada
edgesFile	Nombre del archivo de aristas	String	Entrada
Response	Contiene el grafo generado y un arreglo con los valores de las métricas	List	Salida

El primer paso dentro de esta función es leer los archivos de nodos y aristas, haciendo uso de la función *read.csv*. Para esto se pasan como argumentos, los nombres de los archivos, el separador de los campos, y la opción *stringsAsFactors* se configura en falso para evitar que las cadenas de texto se convierten en factores. Esta información se almacena en dos variables de tipo data frame.

```
cat("Loading nodes and edges file for time ", time, "\n")
nodes <- read.csv(nodesFile, sep = "", stringsAsFactors = FALSE)
edges <- read.csv(edgesFile, sep = "", stringsAsFactors = FALSE)
```

Código 2.13. Lectura de los archivos de nodos y aristas

2.2.2.1.1. Generación del grafo

Con esta información se crea el grafo mediante la función *graph_from_data_frame,* que recibe un data frame con las aristas. En este caso, se toman las columnas *orig* y *dest* que corresponden a cada par de nodos conectados. Además, se toma el id como parte de los metadatos de los vértices y la opción *directed* se configura en falso para indicar que el grafo es no dirigido.

```
net <- graph_from_data_frame(d = edges[, c("orig", "dest")],
                             vertices = nodes$Id,
                             directed = F)
```

Código 2.14. Creación del grafo individual

Una vez creado el grafo, se deben configurar algunas propiedades adicionales. Para diferenciar los nodos que corresponden a vehículos particulares de los buses y transporte público, se asignan los tipos de los vehículos en la propiedad *type* del grafo. Además, se eliminan las etiquetas generadas para los nodos, asignando una cadena de texto vacía.

Para asignar colores a los nodos, de forma que se pueda distinguir visualmente los distintos tipos de autos, se crea una variable llamada *color_vertex* que contiene una paleta de tantos colores como tipos existan mediante la función *terrain.colors.* Luego, se mapean los valores de este vector en la propiedad *color* del grafo.

Como se observó en el archivo de aristas, los valores de la potencia recibida en los nodos tienen un valor negativo ya que está expresado en dBm. Para utilizar estos valores como el peso de cada arista, se deben multiplicar por el factor -1 para convertirlos en valores positivos.

```
V(net)$type = nodes$Type
V(net)$label <- ""
color_vertex <- terrain.colors(max(unique(V(net)$type)))
V(net)$color <- color_vertex[V(net)$type]
E(net)$weight <- -1 * edges[, "strengh"]
```

Código 2.15. Configuración de las propiedades adicionales del grafo

2.2.2.1.2. Obtención de métricas

Una vez que se tienen bien definidas todas las propiedades del grafo, se pueden obtener sus métricas, usando las funciones provistas en la librería *igraph.* Los primeros valores en ser calculados y que resultan muy importantes para describir el grafo, aunque no son métricas de centralidad como tal, son el número de nodos y el número de aristas, mediante

las funciones *vcount* y *ecount,* respectivamente. Estas dos funciones reciben el grafo como único argumento de entrada.

```
## Number of nodes and edges
V.nodes <- vcount(net)
V.edges <- ecount(net)
```

Código 2.16. Cálculo el número de nodos y aristas

Se calcula primero la centralidad basada en grado usando el comando *centr_degree* que recibe como el grafo *net,* el modo en el que se manejan las conexiones (solo entradas, solo salidas o todas), aunque como se está trabajando con un grafo no dirigido, este parámetro es indiferente. También se debe indicar si se deben usar valores normalizados.

En función de este valor, se configura el tamaño de los vértices del grafo y se calculan también el valor máximo, medio y mediana de esta métrica.

```
centr.deg <- centr_degree(net, mode = "all", normalized = T)
deg <- centr.deg[[1]]
deg.centr.value <- centr.deg[[2]]V(net)$size <- deg / max(deg) * 3 + 1
V.deg.mean <- mean(deg)
V.deg.max <- max(deg)
V.deg.median <- median(deg)
```

Código 2.17. Obtención de la centralidad basada en grado

A continuación, se calcula la densidad de aristas utilizando la función *edge density,* que recibe como parámetros el grafo y una bandera que indica si se admiten o no lazos dentro de la red. En este caso puntual, este atributo es falso para indicar que la red no tiene lazos y en caso de haberlos, estas aristas no son tan significativas.

```
V.edge.density <- edge_density(net, 'F')
```

Código 2.18. Obtención de la densidad de aristas

La transitividad se obtiene usando la función *transitivity,* que recibe como parámetros el grafo y una cadena de texto *global,* para indicar que se debe calcular el valor global, es decir, se toma simplemente la tasa de triadas transitivas con respecto a todas las potenciales triadas que se pueden formar.

```
V.transitivity <- transitivity(net, type = "global")
```

Código 2.19. Obtención de la transitividad

Luego, se calcula el diámetro de la red mediante la función *diameter.* Esta función recibe el grafo, una bandera en falso para indicar que este no es dirigido y los pesos de las aristas, que en este caso se configuran como nulos.

Utilizando esta información, se asignan colores a los enlaces o aristas del grafo. Todas las aristas tendrán un color gris y aquellas que formen parte de la ruta más larga (que corresponde al diámetro) se presentarán de color rojo.

```
V.diameter <- diameter(net, directed = F, weights = NA)
E(net)$color <- "gray60"
E(net, path = V.diameter)$color <- "red"
```

Código 2.19. Obtención del diámetro

Para la obtención de la distancia media se dispone de la función *mean_distance,* que también recibe la red y el tipo de grafo. Como ya se ha mencionado para las métricas anteriores, este es un grafo no dirigido, por lo que el parámetro *directed* está en falso.

```
## Mean distance
V.mean.distance <- mean_distance(net, directed = F)
```

Código 2.20. Obtención de la distancia media

Otra de las métricas que se computa es la centralidad basada en intermediación o simplemente conocida como intermediación. Para esto se usa la función *centr_betw,* pasando como argumentos el grafo, una bandera que indica si el grafo es dirigido y otra que indica si se deben utilizar valores normalizados.

```
centr.betw <- centr_betw(net, directed = F, normalized = T)
betw.centr.value <- centr.betw[[2]]
```

Código 2.21. Obtención de la centralidad basada en intermediación

Una vez que se han calculado todas las métricas y con los ajustes realizados en las propiedades y atributos del grafo, se dibuja el grafo y se guarda como una imagen en formato PNG con un nombre descriptivo, tomando en cuenta el escenario (recibido en la variable *folder),* el instante de simulación (recibido en la variable *time).* Para este proceso, se utiliza la función *png* incluida en el paquete *grDevices,* que recibe el nombre del archivo y sus dimensiones. Se utiliza la función *plot* para dibujar el grafo en el archivo y, por último, se utiliza la función *dev.off* para apagar el dispositivo actual y terminar el guardado de la imagen.

```
cat("Creating graph image file for time ", time, "\n")
png(
  file = paste("images/grafo-", folder, "-", time, ".png", sep = ""),
  height = 6910 / 2 ,
  width = 4250 / 2
)
plot(net, layout = l)
dev.off()
```

Código 2.22. Generación de un archivo PNG con el dibujo del grafo

En la Figura 2.9 se presenta un ejemplo de la imagen generada con esta sección de código, donde se puede observar todos los nodos presentes en ese instante de tiempo y las conexiones que lograron establecer entre ellos.

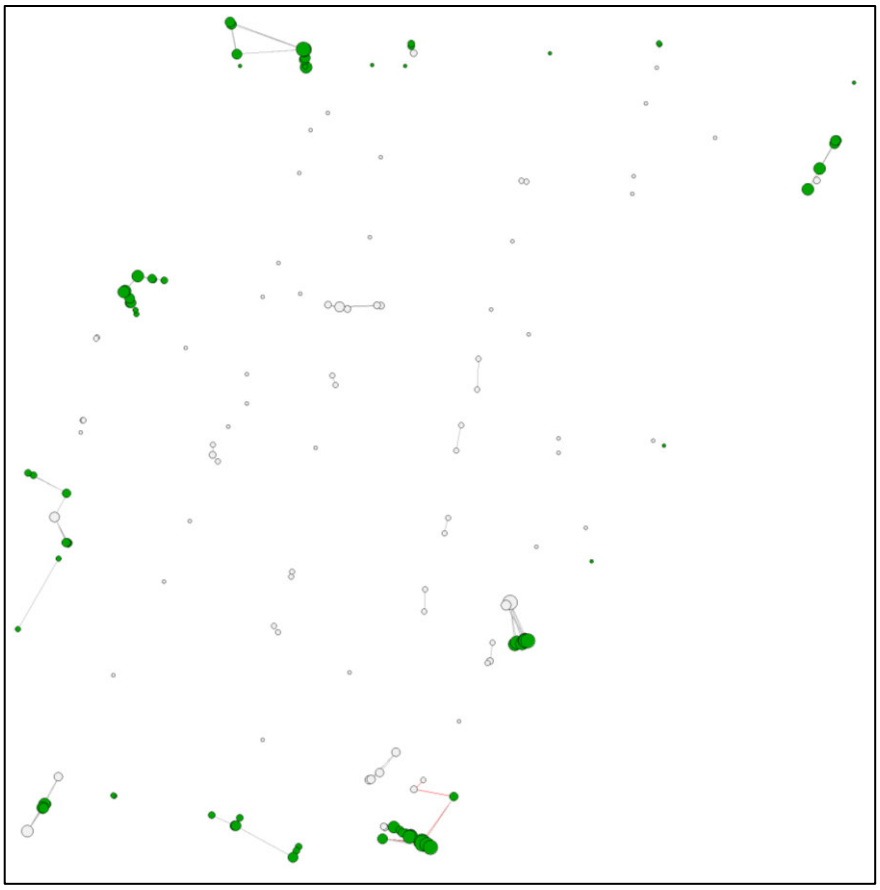

Figura 2.9. Archivo PNG generado con el dibujo del grafo

2.2.2.1.3. Detección de comunidades

Ahora, se calculan el número de comunidades que se forman en la red. Para esto se hace uso de la función *cluster_fast_greedy,* que permite la detección de comunidades o subgrafos dentro del grafo principal, aplicando un método de a optimización directa de una puntuación de modularidad.

Esta función recibe como único argumento el grafo y el resultado se almacena en una variable llamada *cfg*, sobre la cual se pueden aplicar otras funciones del paquete *igraph* para poder analizarla. El número de comunidades se puede obtener a partir de la cantidad de elementos que tiene el vector *cfg,* usando la función *length* que calcula la longitud de un vector dado.

Respecto a las comunidades, otros valores que resultan muy interesantes para el análisis son el modularidad y el *crossing.* El modularidad es una medida de que tan buena es la partición de una red, mientras que *crossing* indica si una arista sirve como conexión de dos o más aristas.

Para obtener el valor de modularidad, se utiliza la función *modularity,* que recibe el vector *cfg* y el *crossing* se obtiene sumando los valores del vector devuelto por la función *crossing* que están marcados como TRUE.

```
cfg <- cluster_fast_greedy(net)
cfg.communities <- length(cfg)
cfg.modularity <- modularity(cfg)
cfg.crossing <- sum(crossing(cfg, net))
```

Código 2.23. Detección de comunidades

El siguiente paso es dibujar estas comunidades en el grafo usando el proceso descrito en el Código 2.22. Se utiliza también un nombre descriptivo que empieza con el prefijo comm, para indicar que se trata del dibujo de las comunidades, seguido del nombre del escenario y el instante de simulación. En la función *plot* se utilizan el vector *cfg* y el grafo como argumentos de entrada.

```
cat("Creating communities image file for time ", time, "\n")
png(
  file = paste("images/comm-", folder, "-", time, ".png", sep = ""),
  height = 6910 / 2 ,
  width = 4250 / 2
)
plot(cfg, net, layout = l)
dev.off()
```

Código 2.24. Generación de un archivo PNG con el dibujo de las comunidades

En la figura 2.10 se presenta un ejemplo de los archivos de imagen generados con el código anterior, donde se puede ver los nodos y las aristas del grafo y además se pintan con colores diferentes cada una de las comunidades formadas en el mismo.

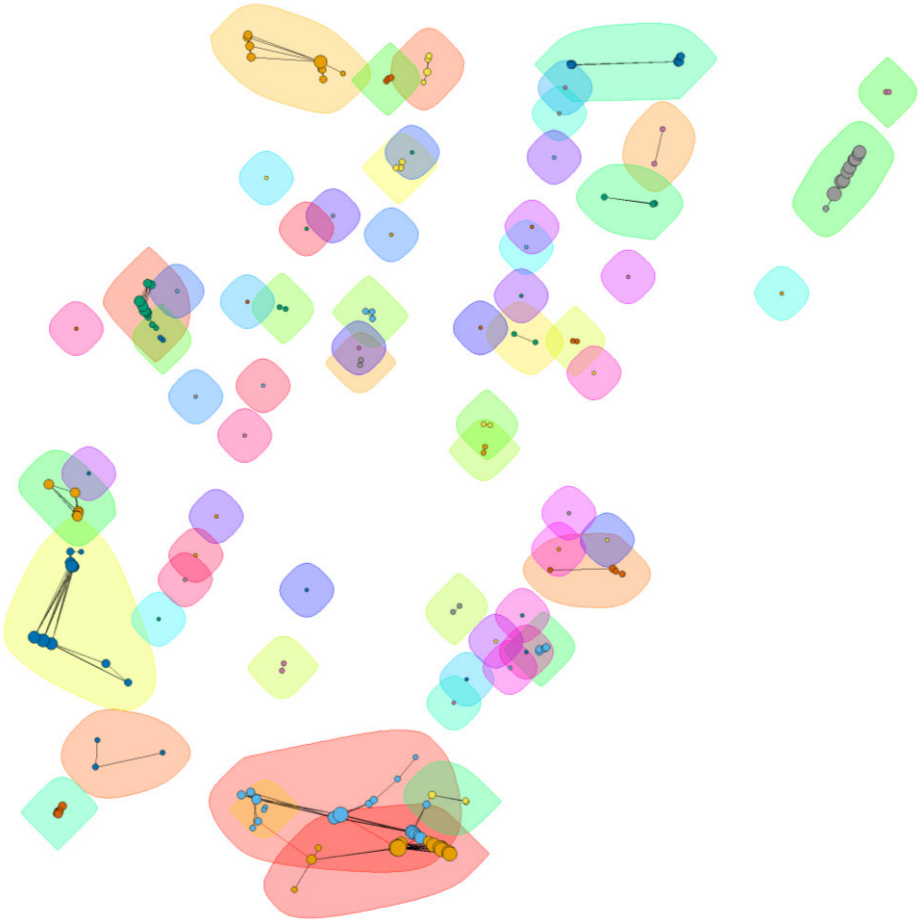

Figura 2.10. Ejemplo de imagen generada con las comunidades

Por último, se crea un data frame con todos los valores calculados para las métricas y la detección de comunidades. Además, se incluyen dos columnas en este data frame que corresponden al escenario e instante de tiempo simulado. En una variable de tipo *list* se agrupa este data frame con el grafo y se devuelve esta lista como salida de la función.

```
newRecord <- data.frame(
  scenario = folder,
  time = time,
  nnondes = V.nodes,
  nedges = V.edges,
  edge.density = V.edge.density,
  deg.mean = V.deg.mean,
  deg.max = V.deg.max,
  deg.median = V.deg.median,
  transitivity = V.transitivity,
  diameter = V.diameter,
  distance.mean = V.mean.distance,
  centr.deg = deg.centr.value,
  centr.betw = betw.centr.value,
  comm.number = cfg.communities,
  comm.modularity = cfg.modularity,
```

```
  comm.crossing = cfg.crossing
)
response = list(net, newRecord)
return(response)
```

Código 2.25. Formación de la respuesta de la función

El código completo del archivo *functions.R* que contiene a la función *get_unit_info* se presenta en el Anexo D.

2.2.2.2. Obtención de datos para todos los instantes de tiempo

Para procesar la información para todos los instantes, se desarrolla otro script de R llamado *porcessData* que itera sobre todo el rango de tiempo llamado a la función descrita en la sección anterior, almacena los valores de las métricas de cada instante en un único vector y con estos se obtienen los valores medios y resumen, para ser utilizados en el análisis de los datos.

El diagrama de flujo de la Figura 2.11 muestra los pasos llevados a cabo para este procesamiento. Se debe procesar el primer registro por separado y no dentro del lazo *for,* para inicializar los vectores que contienen los valores de las métricas y la información de cada grafo. Asimismo, después del procesar todos los instantes de tiempo se calculan dos grafos adicionales y sus métricas.

El primero toma todos los valores individuales y calcula el valor medio de cada uno, mientras que el segundo agrupa todos estos valores para generar un grafo resumen que tiene información de toda la simulación

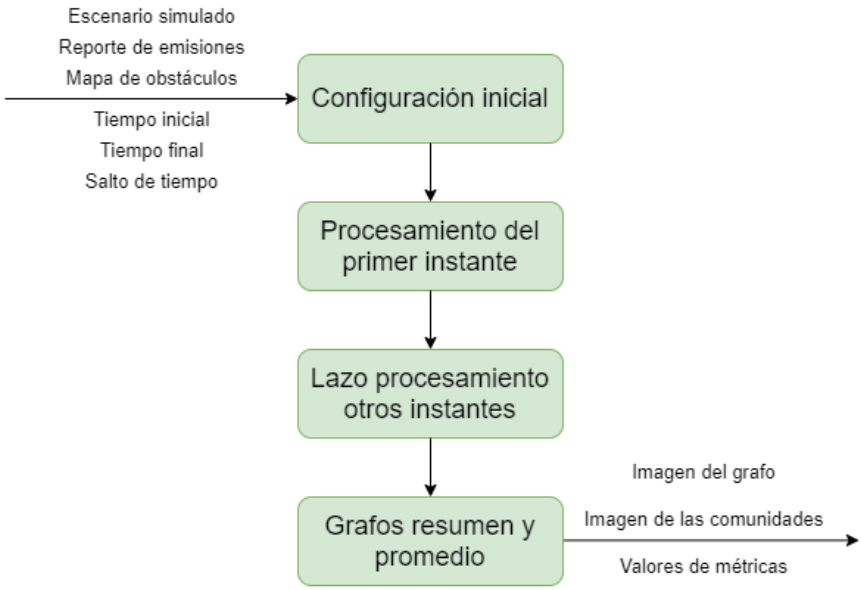

Figura 2.12. Configuración inicial del script

2.2.2.2.1. Configuración inicial

El primer paso en este programa es cargar todas las librerías necesarias para la lectura de archivos, el manejo de grafos y cálculo de sus métricas y la generación de gráficos a partir de estos.

Además, se leen de la línea de comandos todos los argumentos para asignarlos a variables que será utilizadas a lo largo de todo el script. Se recibe el nombre del escenario simulado en la variable *folder,* el reporte de emisiones de SUMO, el archivo de obstáculos en la red, los instantes de inicio, fin y el salto de tiempo.

Se utiliza la función *source* para cargar el código del archivo *functions.R* y se configura el directorio de trabajo en base al nombre del escenario.

```
## Load libraries
cat("Loading required libraries", "\n")
library(readr)
library(igraph)
library(RColorBrewer)
library(ggplot2)
library(modeest)
library(Matrix)

## Get command line arguments
args <- commandArgs(trailingOnly = TRUE)
folder <- args[1]
emm_file <- args[2]
map_file <- args[3]
t_ini <- as.numeric(args[4])
t_fin <- as.numeric(args[5])
step <- as.numeric(args[6])

## Load custom functions from external script
source("functions.R")

## Set current working directory
wd <- paste("D:\\Tesis\\Procesamiento\\", folder, sep = "")
cat("Setting working directory to ", wd, "\n")
setwd(wd)
```

Código 2.26. Configuración inicial del script

2.2.2.2.2. Procesamiento del primer instante de tiempo

En esta sección, se realiza el procesamiento del primer registro, ya que los valores calculados para cada instante de tiempo se almacenan en arreglos y estos deben ser inicializados con algún valor, caso contrario no podrán ser usados en el lazo *for.*

Primero, se forman las cadenas de texto con los nombres de los archivos de nodos y aristas tomando como referencia el instante de tiempo inicial almacenado en la variable *t_ini.* Con estos nombres se llama a la función *get_unit_data,* que retorna la lista con el grafo y los valores de las métricas. Estos valores también se almacenan como un archivo con extensión RData.

```
## Load initial time files
cat("Processing initial record. Initial time = ", t_ini, "\n")
nodesFile <- paste("data/nodes", t_ini, ".txt", sep = "")
edgesFile <- paste("data/edges", t_ini, ".txt", sep = "")

## Use imported function to generate the individual graph and
iniList <- get_unit_info(folder, t_ini, nodesFile, edgesFile)

## First returned value is the individual graph
iniNet <- iniList[[1]]

## Second returned value is the list of metrics
iniMetrics <- iniList[[2]]

## Save variables in RData format using a descriptive name (folder-time)
## We save the list of metrics and graph for initial time
cat("Saving Rdata for initial time ", t_ini, "\n")
iniRecord <- paste(folder, t_ini, sep = "-")
assign(iniRecord, iniMetrics)
assign(paste("G", folder, t_ini, sep = "."), iniNet)
save(list = c(paste(folder, t_ini, sep = "-"),
              paste("G", folder, t_ini, sep = ".")),
    file = paste("Rdata/", folder, "-", t_ini, ".RData", sep = ""))

## Remove variables from workspace
rm(list=c(paste(folder,t_ini,sep="-"),paste("G",folder,t_ini,sep = ".")))
```

Código 2.27. Procesamiento del primer registro

El siguiente paso es eliminar algunas propiedades irrelevantes del grafo, color de nodos y aristas, tamaño de los nodos y etiquetas, utilizando las funciones *delete_vertex_attr* y *delete_edge_attr.*

Se inicializan las variables *individualRecords* y *GSummary* con los valores obtenidos de la función *get_unit_info.* La primera de estas variables corresponde a un data frame que guardará en cada fila las métricas calculadas para cada instante de tiempo. Por su parte, la variable *GSummary* unirá cada grafo individual para tener al finassl un grafo con la información de toda la simulación

Por último, se eliminan las variables que ya no serán utilizadas para evitar que el espacio de trabajo se llene de tantos objetos innecesarios.

```
## Delete color, size and label vertex attributes from net
iniNet <- delete_vertex_attr(iniNet, "color")
iniNet <- delete_vertex_attr(iniNet, "size")
iniNet <- delete_vertex_attr(iniNet, "label")

## Delete color edge attribute from net
iniNet <- delete_edge_attr(iniNet, "color")

## This variable has all individual metrics as a data frame
individualRecords <- iniMetrics

## This variable has all individual graphs to get a summarized graph at the end
GSummary<-iniNet

## Delete initial variables from environment
rm(list=c("iniList","iniNet","iniMetrics","iniRecord"))
```

Código 2.28. Procesamiento del primer registro

2.2.2.2.3. Procesamiento del resto de instantes de tiempo

Con las variables *individualRecords* y *GSummary* inicializadas, se utiliza un lazo *for* para recorrer el resto de los instantes de tiempo. La lógica dentro este bucle es la misma que se utilizó para el procesamiento del primer registro: se forman los nombres de los archivos de nodos y aristas en función del instante actual, se invoca a la función *get_unit_info* y se almacenan los valores obtenidos en un archivo.

La primera diferencia está al almacenar la información de las métricas en la variable *individualRecords,* pues se utiliza la función *rbind* para agregar los valores calculados en una nueva fila de este data frame.

```
individualRecords <- rbind(individualRecords,metrics)
```

Código 2.29. Inserción de los valores calculados en una nueva fila del data frame

Para el caso del grafo, se utiliza la función *graph.union* que permite combinar varios grafo en uno solo, pasando como argumentos el grafo *GSummary,* el grafo calculado para el intervalo actual y una bandera *byName* con el valor TRUE para indicar que la unión se realiza en función de los nombres simbólicos de los nodos y no en función de sus identificadores numéricos.

```
GSummary<-graph.union(GSummary,net,byname = T)
```

Código 2.30. Unión de los grafos

Se realiza un procesamiento adicional con las propiedades *weight* de las aristas y *type* de los nodos, puesto que después de la unión, se disponen de dos pesos para cada arista y dos tipos para cada nodo.

Primero se reemplazan los valores nulos de las columnas por ceros. En el caso de las aristas el peso total de una resulta de la suma de las dos columnas *weight* que corresponden a los pesos de los grafos *Gsummary* y del intervalo actual.

Para el caso de los nodos, la propiedad *type* toma el valor máximo entre los dos valores disponibles, que nuevamente corresponde al tipo de vehículo en el grafo *Gsummary Gsummary* y en el grafo del instante de tiempo actual.

```
E(GSummary)$'weight_1'[is.na(E(GSummary)$"weight_1")]<-0
E(GSummary)$'weight_2'[is.na(E(GSummary)$"weight_2")]<-0
E(GSummary)$weight<-E(GSummary)$weight_1+E(GSummary)$weight_2
GSummary<-delete_edge_attr(GSummary, "weight_1")
GSummary<-delete_edge_attr(GSummary, "weight_2")

V(GSummary)$"type_1"[is.na(V(GSummary)$"type_1")]<-0
V(GSummary)$"type_2"[is.na(V(GSummary)$"type_2")]<-0
V(GSummary)$"type"<-max(V(GSummary)$"type_1",V(GSummary)$"type_2")
GSummary<-delete_vertex_attr(GSummary, "type_1")
GSummary<-delete_vertex_attr(GSummary, "type_2")
```

Código 2.31. Procesamiento luego de la unión de los grafos

Al finalizar el bucle, el data frame *individualRecords* tiene en cada fila los valores de las métricas de todos saltos de tiempo dentro del rango, mientras que el grafo *Gsumamary* junta toda la información de estos intervalos.

2.2.2.3. Obtención de valores medios y resumen

Con los valores obtenidos en la sección anterior, se pueden obtener las métricas en base al grafo resumen denominado *Gsummary* siguiendo el mismo proceso que la función *get_unit_info,* pero pasando como red este grafo.

Por otra parte, se pueden calcular los valores medios en función de la información de cada intervalo, utilizando *lapply* que permite aplicar la función *mean* al data frame *individualRecords.* Con esto, se obtiene la media aritmética de cada columna, es decir, se calcula el valor medio para cada métrica.

```
averageRecord <- as.data.frame(lapply(individualRecords[,-c(1,2)], mean))
averageRecord$scenario <- folder
averageRecord$type <- "Averages"
```

Código 2.32. Obtención de la media aritmética

2.2.3. SCRIPT MAESTRO

Para ejecutar todos los programas descritos en las secciones anteriores de una forma más organizada y automatizada, se crea un programa maestro que invoca a los scripts para la obtención de la información de nodos y aristas a partir del reporte de emisiones de SUMO y la generación de los grafos hasta llegar al cálculo de las métricas.

Tomando en cuenta que, para la realización de este trabajo se utiliza una máquina con sistema operativo Windows y la variedad de comandos que pueden ser ejecutados desde la línea de comandos, se desarrolla el script como un archivo BATCH. Este script puede ser modificado para ser usado en otros sistemas operativos como Linux.

A fin de tener los directorios de los escenarios simulados más organizados, se maneja una estructura dentro de estos dividida en carpetas: *data, RData* e *images.* Dentro de la carpeta data se almacenan todos los archivos de nodos y aristas, en *RData* se encuentran todas las variables exportadas en los scripts de R y en *images* se guardan las imágenes de los grafos y comunidades de cada intervalo.

La primera parte consiste en tomar la hora de inicio de la ejecución del programa que será utilizada para calcular la duración total, configurar el directorio de trabajo y verificar si los subdirectorios existen, para borrar todo su contenido.

```
:: Save start time in a local variable
set start=%time%

:: Create path string concatenating the first command line arg with a base path
set @wd=D:\Tesis\Procesamiento\%1

:: Delete existing folders in the working directory
echo Checking if folders exist
if exist %@wd%\data (echo Deleting data folder && rmdir /S /Q %@wd%\data)
if exist %@wd%\Rdata (echo Deleting Rdata folder && rmdir /S /Q %@wd%\Rdata)
if exist %@wd%\images (echo Deleting images folder && rmdir /S /Q %@wd%\images)

:: Create empty folders for saving nodes, edges, RData and created images from
echo Creating data folder && mkdir %@wd%\data
echo Creating Rdata folder && mkdir %@wd%\Rdata
echo Creating images folder && mkdir %@wd%\images
echo Folders are ready
```

Código 2.33. Configuraciones iniciales del script maestro

El siguiente paso consiste en invocar al programa *getData.R* para obtener los nodos y aristas. Simplemente se utiliza el comando Rscript y a continuación se especifica el nombre del programa, aunque previamente debe estar configurada la ruta de R dentro la variable de entorno *path.* En el Código 2.34 se puede notar que luego del nombre del programa se

utiliza la opción *%** que permite transferir todos los argumentos recibidos como argumentos del script de R.

```
:: Execute script to generate nodes and edges files
echo Executing getData.R
Rscript getData.R %*
```

Código 2.34. Llamada al programa getData.R

Para el procesamiento de los nodos y aristas, obtención de los grafos y cálculo de las métricas se utiliza el mismo procedimiento para invocar al programa *processData.R*

```
:: Execute script to process data, generate graphs and calculate metrics
echo Executing processData.Rem
Rscript processData.R %*
```

Código 2.35. Llamada al programa processData.R

Una vez que finaliza todo el procesamiento de los datos, se toma la hora de finalización y con la diferencia con la hora de inicio corresponderá a la duración de esta ejecución. Para realizar esta resta se dividen estos tiempos en horas, minutos, segundos y milisegundos y se realiza un ajuste en los casos que la diferencia resulte un valor negativo. Para finalizar se muestra en la consola un mensaje con este tiempo transcurrido.

```
:: Save end time in a local variable
set end=%time%

:: Aux variable to iterate times
set options="tokens=1-4 delims=:.,"

:: Get hours, minutes, seconds and miliseconds from start and end times
for /f %options% %%a in ("%start%") do set start_h=%%a&set /a start_m=100%%b %%
100&set /a start_s=100%%c %% 100&set /a start_ms=100%%d %% 100
for /f %options% %%a in ("%end%") do set end_h=%%a&set /a end_m=100%%b %% 100&set
/a end_s=100%%c %% 100&set /a end_ms=100%%d %% 100

:: Calculate difference between end and start time for hours, minutes, seconds and
miliseconds
set /a hours=%end_h%-%start_h%
set /a mins=%end_m%-%start_m%
set /a secs=%end_s%-%start_s%
set /a ms=%end_ms%-%start_ms%

:: Adjust values if difference was a negative value
if %ms% lss 0 set /a secs = %secs% - 1 & set /a ms = 100%ms%
if %secs% lss 0 set /a mins = %mins% - 1 & set /a secs = 60%secs%
if %mins% lss 0 set /a hours = %hours% - 1 & set /a mins = 60%mins%
if %hours% lss 0 set /a hours = 24%hours%
if 1%ms% lss 100 set ms=0%ms%
```

```
:: Show total execution time
set /a totalsecs = %hours%*3600 + %mins%*60 + %secs%
echo Excution time: %hours%:%mins%:%secs%.%ms% [%totalsecs%.%ms% sec(s) in total]
```

Código 2.36. Cálculo de la duración de la ejecución del script maestro

En la Figura 2.13 se muestra un ejemplo de las salidas en consola al ejecutar este programa maestro.

```
Creating graph image file for time  4440
Creating communities image file for time  4440
Saving Rdata for time  4440
Loading nodes and edges file for time  4470
Creating the base grap for time  4470
Calculating metrics for time  4470
Creating graph image file for time  4470
Creating communities image file for time  4470
Saving Rdata for time  4470
Loading nodes and edges file for time  4500
Creating the base grap for time  4500
Calculating metrics for time  4500
Creating graph image file for time  4500
Creating communities image file for time  4500
Saving Rdata for time  4500
Calculating number of nodes and edges
Calculating summary degree centrality
Calculating summary edge density
Calculating summary transitivity
Calculating summary diameter
Calculating summary mean distance
Calculating summary betweenness
Calculating summary eigenvector
Processing communities for summary
Getting average statistics
Saving summary Rdata
Geetting adjacency matrix
Drawing plot Time 100
Drawing plot Time 150
Drawing plot Time 200
Drawing plot Time 250
Drawing plot Density 100
Drawing plot Density 150
Drawing plot Density 200
Drawing plot Density 250
Excution time: 0:2:6.37 [126.37 sec(s) in total]
```

Figura 2.13. Ejemplo de ejecución del script maestro

En el Anexo E se muestra el código completo de este programa maestro, mientras que en el Anexo F se detalla la organización del repositorio que contine todos los programas descritos en esta sección y los escenarios simulados en este trabajo.

3. RESULTADOS Y DISCUSIÓN

A continuación, se presentan los resultados obtenidos al finalizar la ejecución de los programas y código descritos en la sección anterior. Para analizar cada métrica se toman en cuenta todos los parámetros que varían entre los distintos casos propuestos (probabilidad, intervalo de reenrutamiento y densidad vehicular) y se generan gráficos en base a los valores promedios de todos los instantes de la simulación y el grafo resumen. Con estos gráficos es posible establecer las diferencias en el comportamiento de los vehículos en escenarios con reenrutamiento activado y aquellos que no lo tienen.

Finalmente, se obtiene información adicional

3.1. NÚMERO DE NODOS

Como se mencionó en el primer capítulo, el número de nodos no es una métrica como tal, pero este valor es muy útil para conocer el tamaño de la red. Este número está muy relacionado con la densidad vehicular, puesto que, en los escenarios con densidad baja, se simula una menor cantidad de autos y evidentemente, el número de nodos será menor en estos casos.

Un primer análisis consiste en fijar un intervalo de reenrutamiento y dibujar el número de nodos para cada densidad. En la Figura 3.1, los valores promedio para un intervalo de 3 minutos son prácticamente los mismos sin importar la probabilidad de reenrutamiento y por esta razón todas las líneas se presentan sobrepuestas. Se puede observar además que mientras aumenta la densidad vehicular, aumenta el número de nodos presentes en la red.

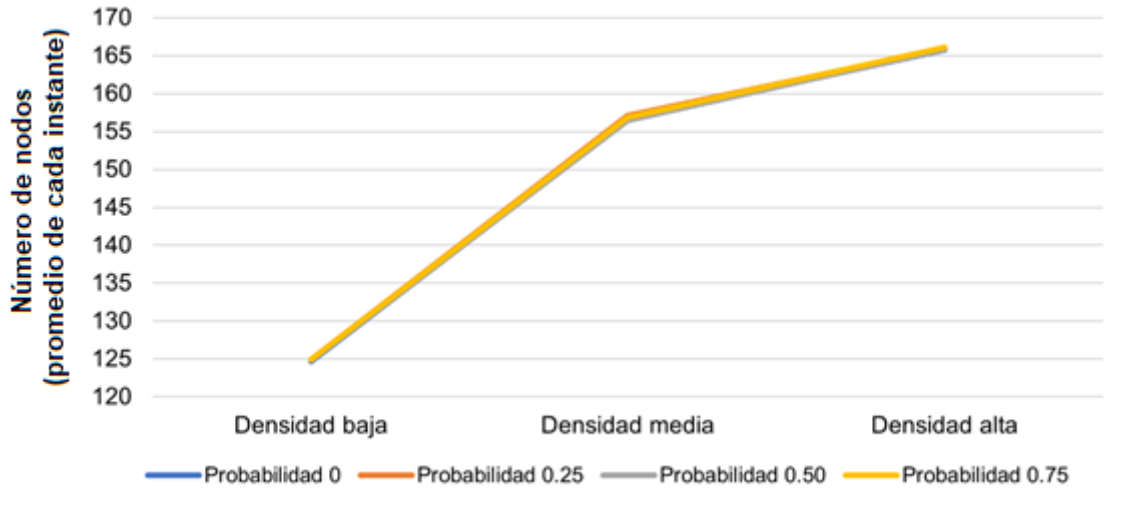

Figura 3.1. Número de nodos para un intervalo de reenrutamiento fijo (valor promedio)

Si se toman en cuenta los valores calculados a partir del grafo resumen, se puede notar que el comportamiento es muy similar al anterior y el menor número de nodos se presenta en los escenarios con densidad vehicular baja.

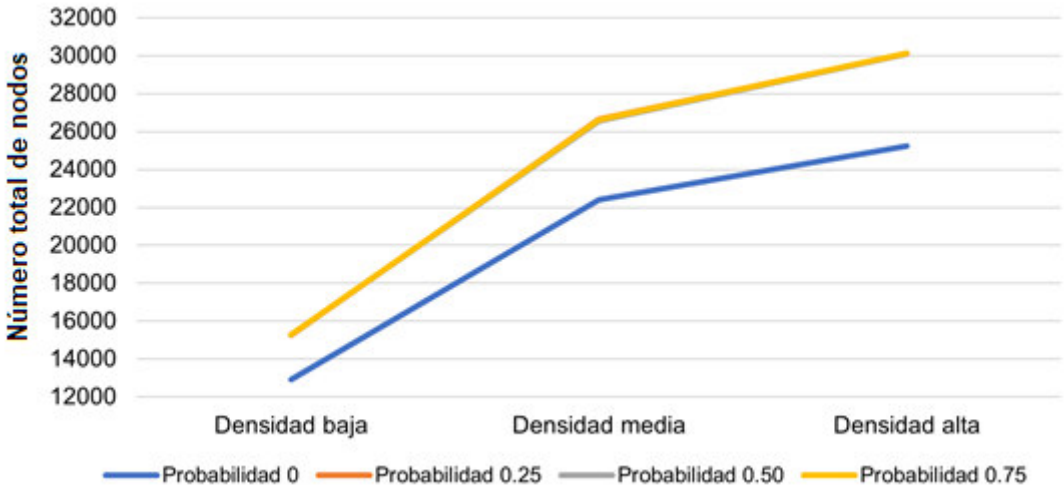

Figura 3.2. Número de nodos para un intervalo de reenrutamiento fijo (resumen)

Otra forma de estudiar esta métrica es establecer una probabilidad y dibujar los valores para cada intervalo de reenrutamiento. En la Figura 3.3 se puede observar que. para cada densidad, el número de nodos es muy parecido sin importar el intervalo escogido. Esto significa que los vehículos cambian la frecuencia con que buscan una nueva ruta más corta hacia su destino, pero desde el punto de vista de la red, el número de autos presentes no varía.

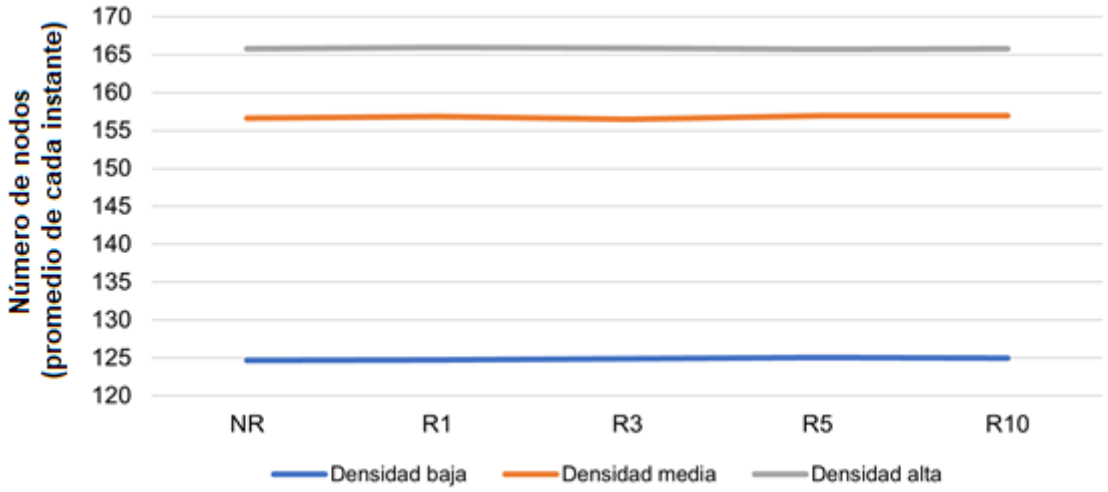

Figura 3.3. Número de nodos para una probabilidad de reenrutamiento fija (promedio)

Si se realiza el mismo análisis en función de los valores resumen, el comportamiento es bastante similar para los diferentes intervalos de reenrutamiento. La principal diferencia

respecto al caso anterior se da en los escenarios sin reenrutamiento habilitado, donde el número de nodos es menor.

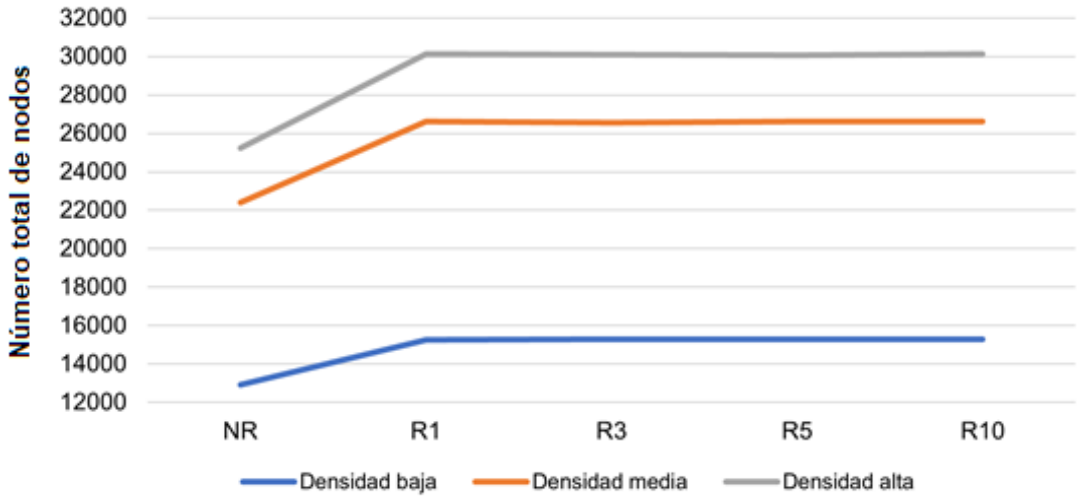

Figura 3.4. Número de nodos para una probabilidad de reenrutamiento fija (resumen)

Una última forma de análisis es establecer un valor fijo para la densidad vehicular y comparar los valores obtenidos para las distintas probabilidades. En la Figura 3.5 se observar que, para el grafo promedio el número de nodos disminuye cuando la probabilidad tiende al 50%. Esto significa que existen menos nodos dentro de la red cuando la probabilidad no está en ninguno de los extremos.

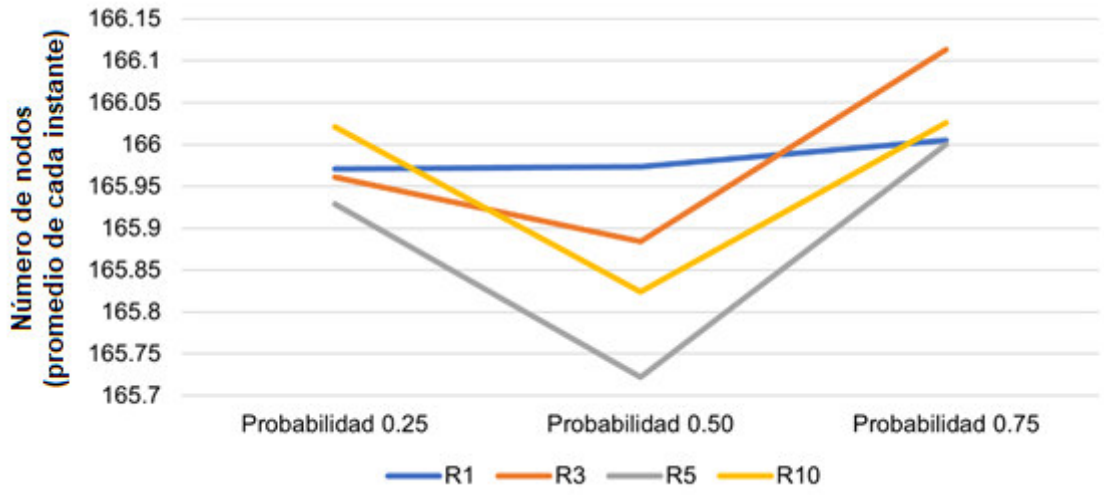

Figura 3.5. Número de nodos para una densidad fija

Cuando la probabilidad es muy baja, pocos vehículos cambian sus rutas y se genera mayor congestión, por lo que el número de autos en cada instante es alto. De la misma forma, cuando la probabilidad de reenrutamiento es muy alta, casi todos los vehículos están

buscando nuevas rutas hacia su destino, lo que ocasiona un cierto nivel de congestión vehicular y, por lo tanto, el número de nodos también aumenta.

3.2. CENTRALIDAD BASADA EN GRADO

La centralidad basada en grado indica el número de aristas que conectan un nodo y con esto la importancia que tiene un nodo respecto a sus vecinos. Primero se analizan los valores obtenidos para las distintas densidades considerando una probabilidad de reenrutamiento fija.

Cuando la densidad es baja, el grado de centralidad también es bajo ya que existen menos vehículos en la red y no se forman tantas conexiones entre los diferentes nodos. Para los escenarios de densidades media y alta, los valores son relativamente más altos.

En la Figura 3.6 se muestran los resultados obtenidos para una probabilidad de 25% y al no existir muchos cambios en las rutas de los vehículos, las líneas están sobrepuestas indicando un comportamiento similar a pesar de que se varíe el intervalo con el cual los autos buscan una nueva ruta.

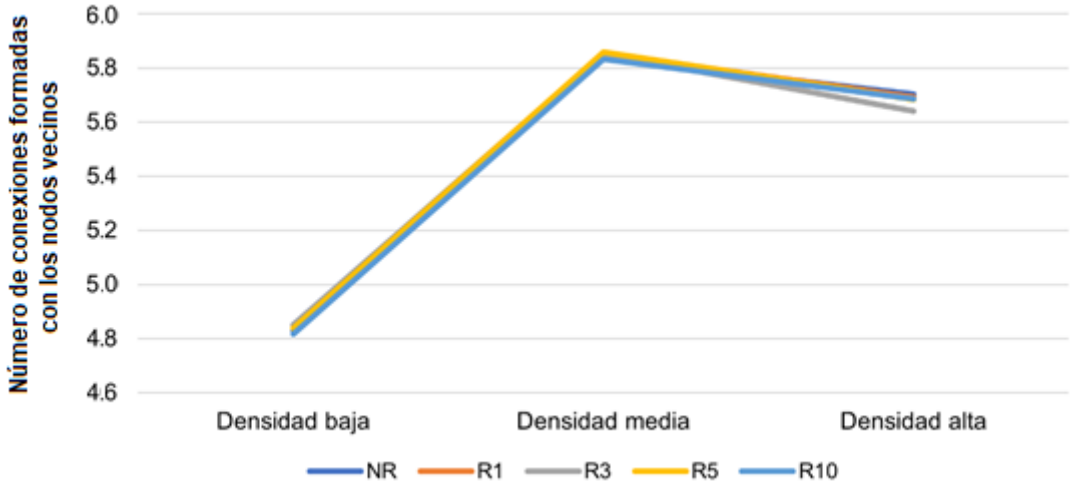

Figura 3.6. Centralidad basada en grado para probabilidad de 25%

Cuando la probabilidad empieza a incrementar, como el caso de la Figura 3.7, donde se toman los valores cuando la probabilidad es 50%, las líneas para cada intervalo empiezan a distinguirse, por lo que se deduce que a medida que incrementa la cantidad de autos con reenrutamiento activado, los períodos para buscar nuevas rutas empiezan a cobrar más importancia.

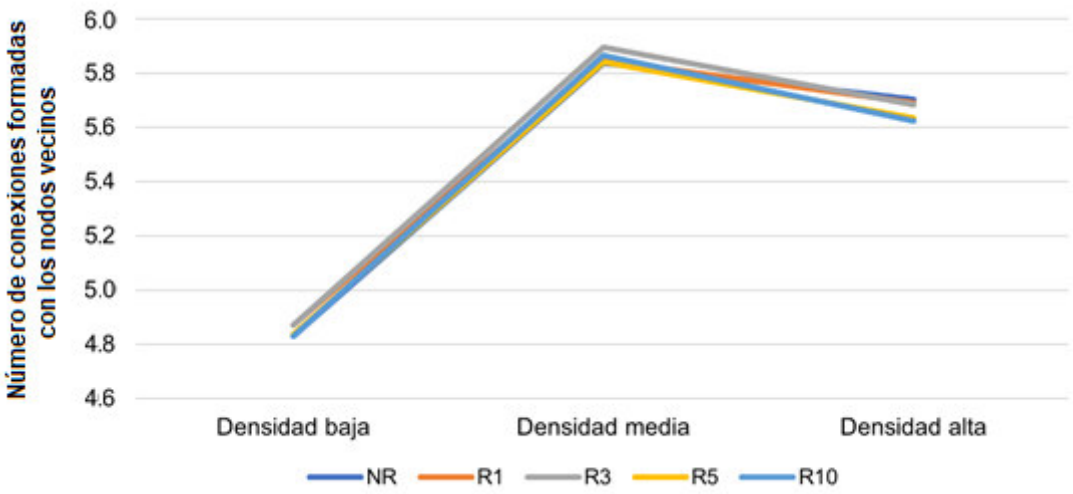

Figura 3.7. Centralidad basada en grado para probabilidad de 50%

Para el caso con la probabilidad más alta (75%) las diferencias son más notorias, y los intervalos que presentan un mejor rendimiento son 3 y 5 minutos. Para el caso del período igual a 1 minuto, los valores son un poco más altos, porque los vehículos están constantemente calculando una nueva ruta más corta hacia su destino, lo que se traduce bajo ciertas circunstancias en mayor congestión y, por ende, un mayor grado de centralidad. Para el intervalo de 10 minutos el comportamiento es muy similar al escenario cuando no se tiene activado el reenrutamiento, es decir, que no existen muchos cambios de rutas durante la simulación.

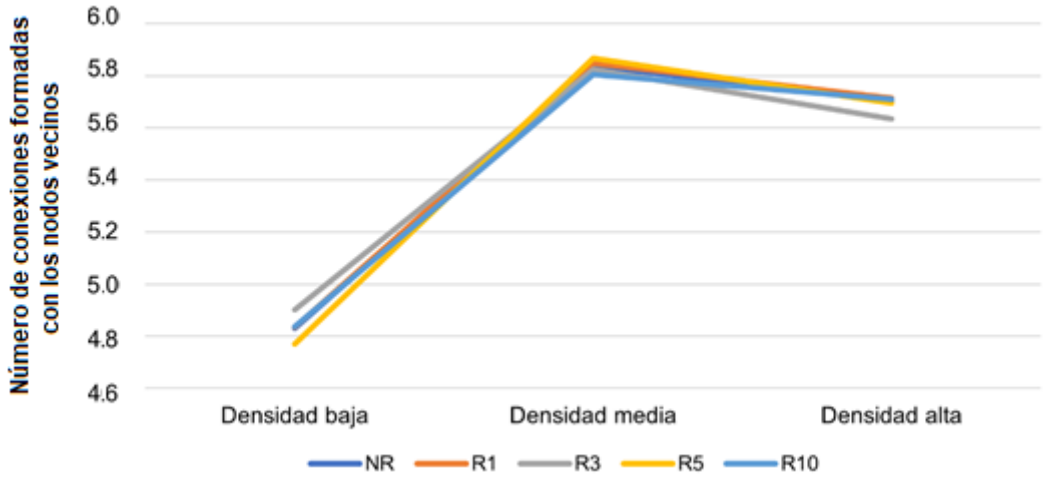

Figura 3.8. Centralidad basada en grado para probabilidad de 75%

Para estudiar esta métrica también se puede fijar el período de reenrutamiento y variar los escenarios en densidad y probabilidad. Nuevamente, los valores que presentan un mejor comportamiento son los intervalos de 3 y 5 minutos. Además, se puede notar que, para los casos con una densidad baja, el grado de centralidad es mucho mayor que para las

densidades media y baja, lo que significa que, en estos casos, los nodos tienen una mayor importancia respecto a sus vecinos.

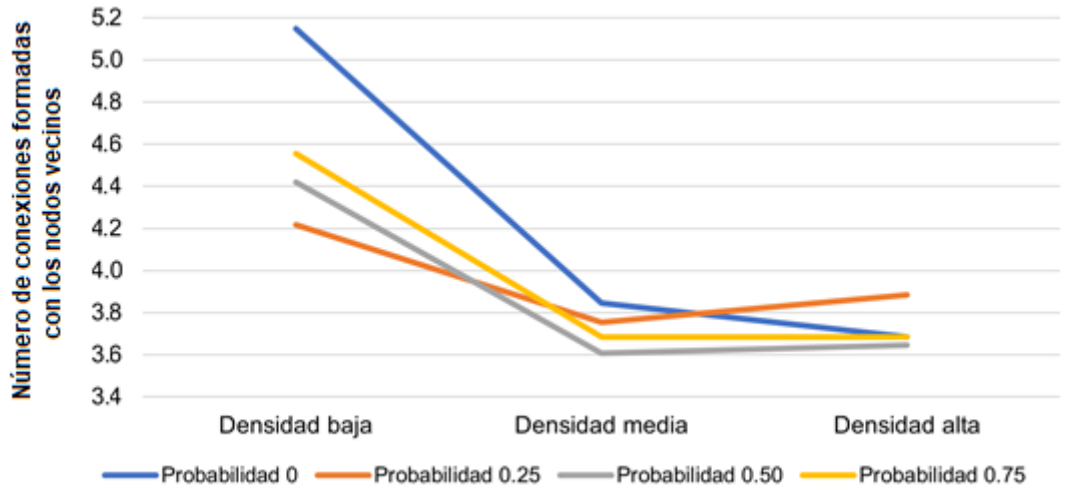

Figura 3.9. Centralidad basada en grado para intervalo de 1 minuto

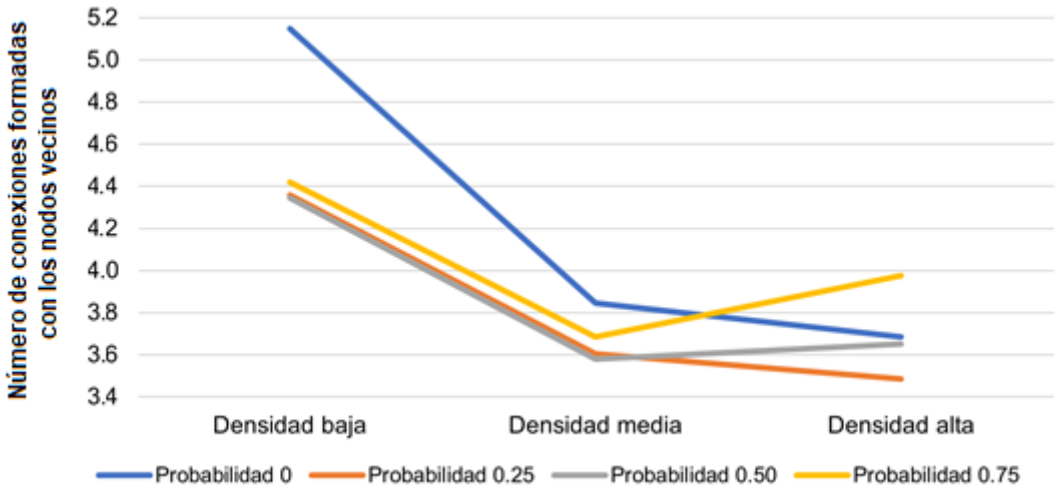

Figura 3.10. Centralidad basada en grado para intervalo de 3 minutos

Cuando se incrementa el intervalo de reenrutamiento, cobra más importancia la probabilidad elegida, puesto que, si los vehículos se dejan pasar más tiempo para tratar de calcular una nueva ruta más corta, esto se puede compensar si se aumenta la probabilidad y así disminuir el impacto. En las Figuras 3.11 y 3.12 se presentan los resultados del grado de centralidad para los intervalos de 5 y 10 minutos, donde la curva de la probabilidad más alta presenta el mejor comportamiento.

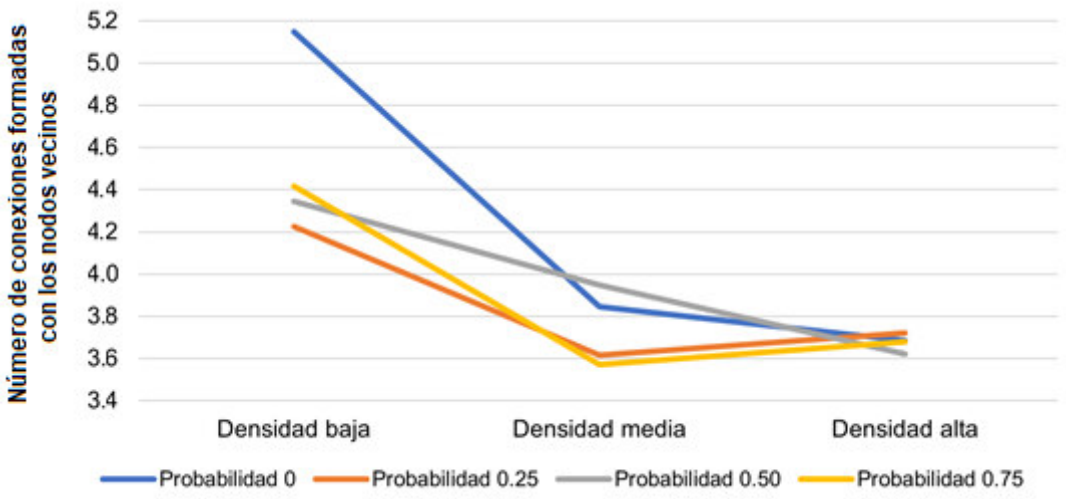

Figura 3.11. Centralidad basada en grado para intervalo de 5 minutos

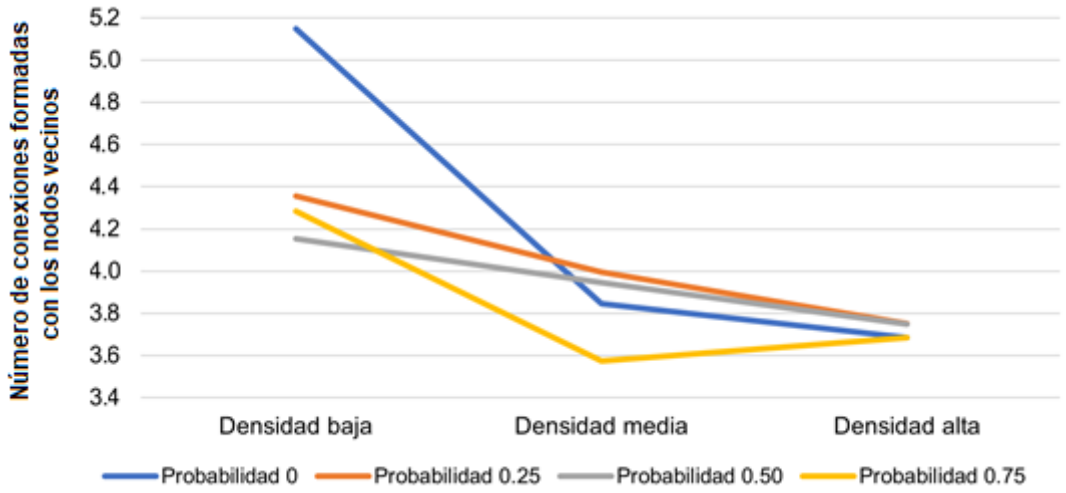

Figura 3.12. Centralidad basada en grado para intervalo de 10 minutos.

Luego de revisar el comportamiento de esta métrica en los diferentes escenarios, se puede concluir que existe una gran relación entre la centralidad basada en grado y el número de nodos presentes en la red, puesto que ante una mayor densidad de vehículos se incrementa el número de conexiones entre ellos y por tanto los nodos tienen un grado de centralidad más alto.

3.3. INTERMEDIACIÓN

Al ser la intermediación es una medida que indica la cantidad de veces que un nodo actúa como puente en la conexión de otro par de nodos, está muy relacionada con la centralidad basada en grado. Si un nodo tiene un grado de centralidad alto, es más importante dentro de la red y se espera que sirva más veces como puente.

Se revisan los mismos escenarios que la sección anterior para verificar que efectivamente el comportamiento de estas dos métricas es bastante parecido. Para la probabilidad más baja (25%) las curvas de todos los intervalos de reenrutamiento están alineadas y casi sobrepuestas.

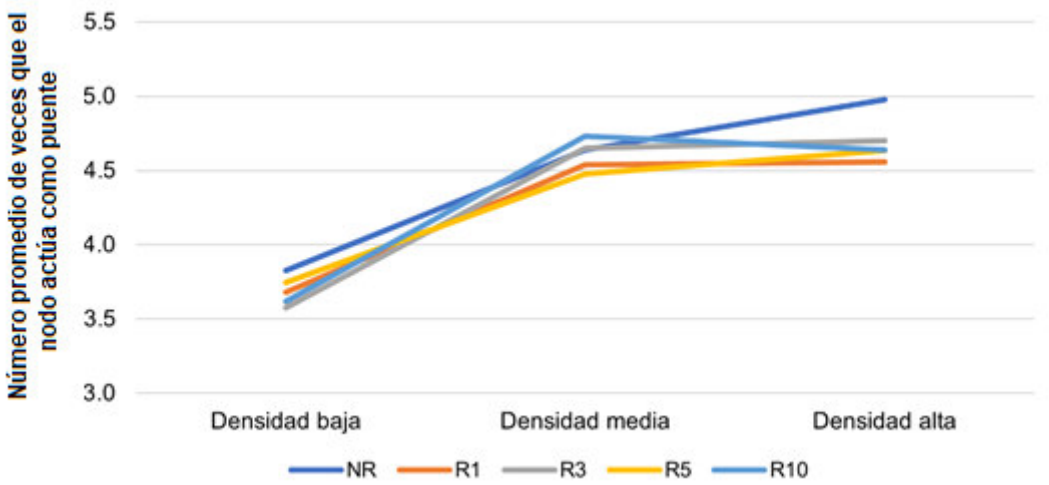

Figura 3.13. Intermediación para probabilidad de 50%

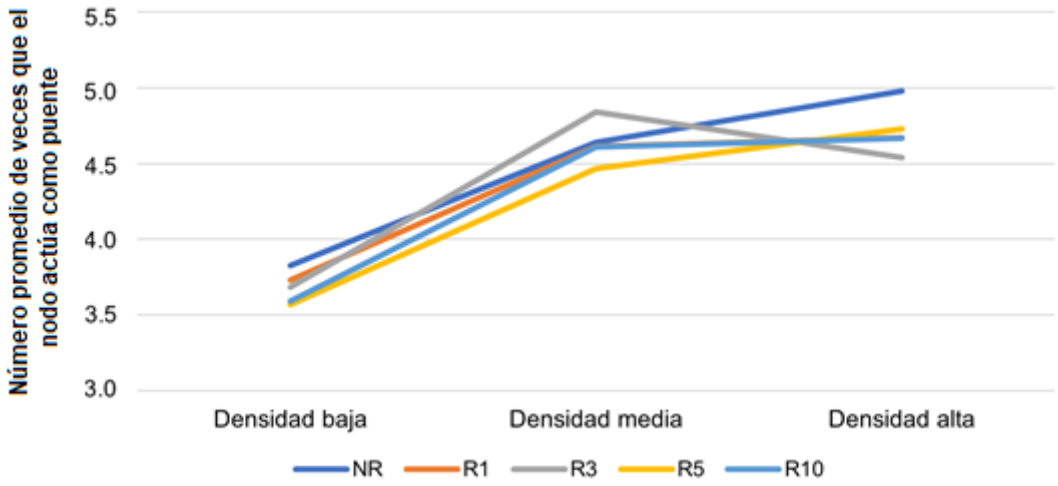

Figura 3.14. Intermediación para probabilidad de 50%

Para las probabilidades de 50% y 75% las líneas se distinguen un poco más entre sí y los valores de la intermediación se incrementan levemente respecto a la probabilidad más baja.

Nuevamente, los intervalos de 3 y 5 minutos presentan un mejor comportamiento, debido a que producen una menor congestión y por lo tanto se reduce el número de conexiones formas entre los vehículos con esto el valor de intermediación es más bajo.

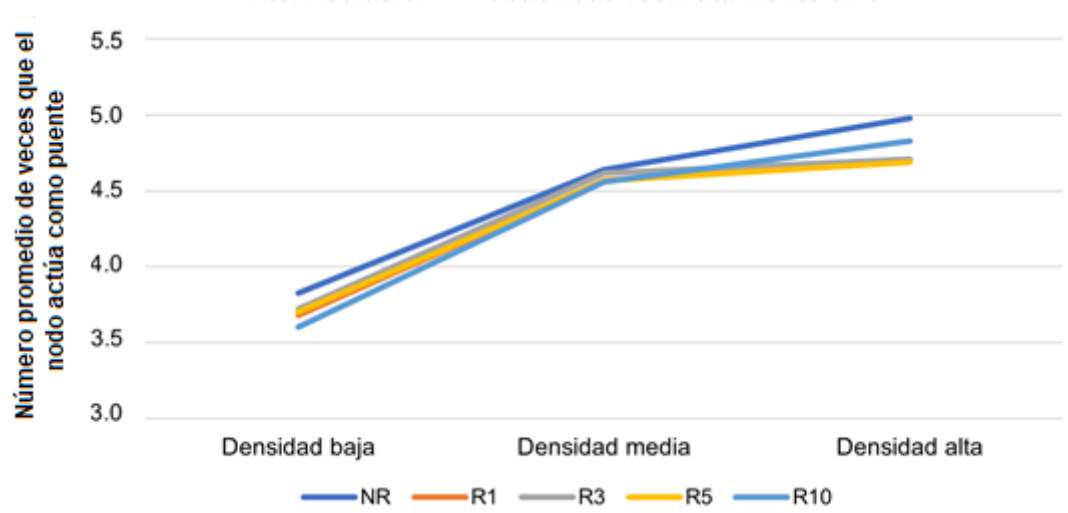

Figura 3.15. Intermediación para probabilidad de 75%

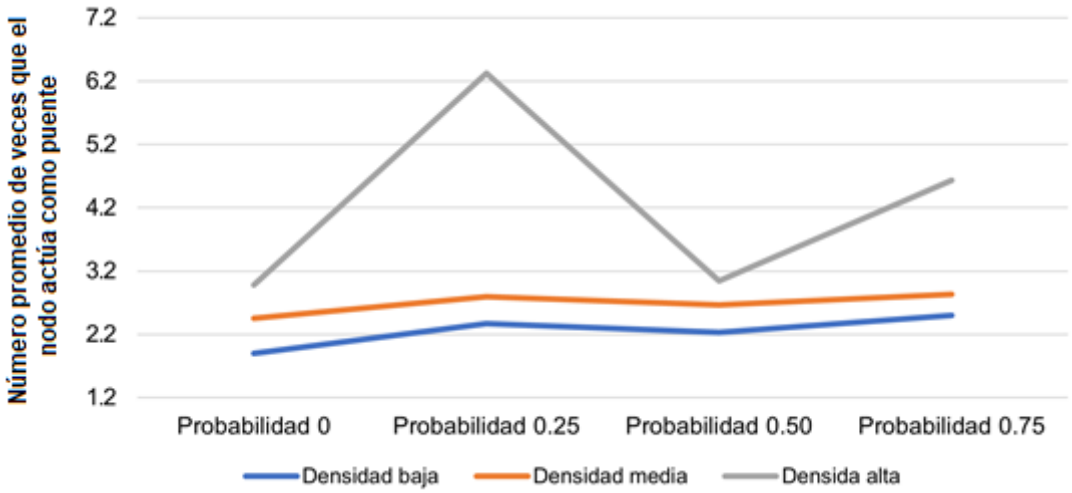

Figura 3.16. Intermediación para intervalo de 1 minuto

La segunda forma de analizar esta métrica es fijar un intervalo y variar la densidad y las probabilidades. Para un intervalo de 1 minuto, las densidades baja y media tienen un comportamiento similar sin importar la probabilidad elegida. No obstante, la densidad alta presenta un valor muy alto para la probabilidad de 25%, es decir, en este escenario existen muchos vehículos dentro de la red y un pequeño grupo empieza a tomar nuevas rutas constantemente (cada 1 minuto), por lo que al moverse forman más conexiones con el resto y se incrementa el valor de la intermediación para este caso. Este comportamiento se puede evidenciar en la Figura 3.16.

Por su parte, en la Figura 3.17 se muestra el mismo estudio para el intervalo de 3 minutos. Con la densidad media se evidencia un mejor comportamiento para la probabilidad de 50%, aunque esta misma probabilidad presenta el peor valor cuando existe una densidad

vehicular alta. Para los escenarios con una densidad baja el comportamiento es similar para los distintos valores de probabilidad.

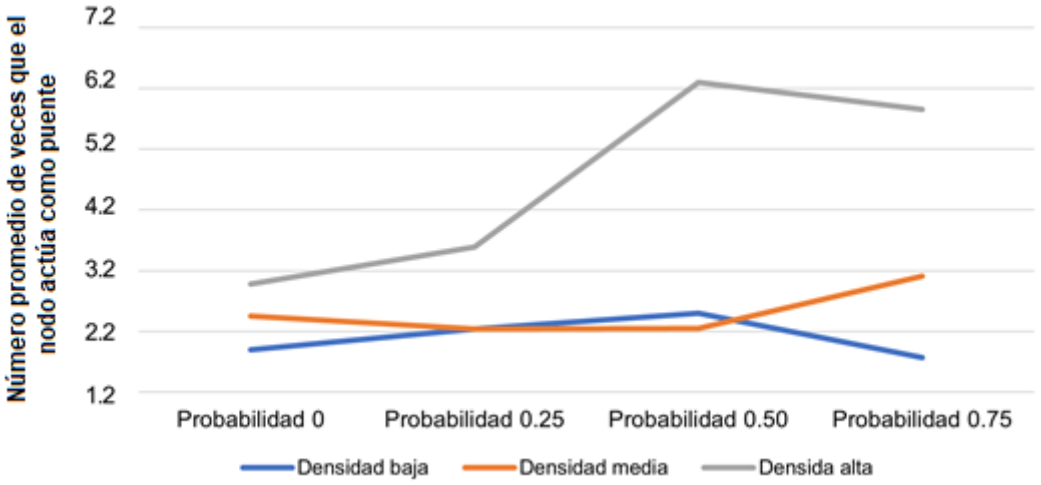

Figura 3.17. Intermediación para intervalo de 3 minutos

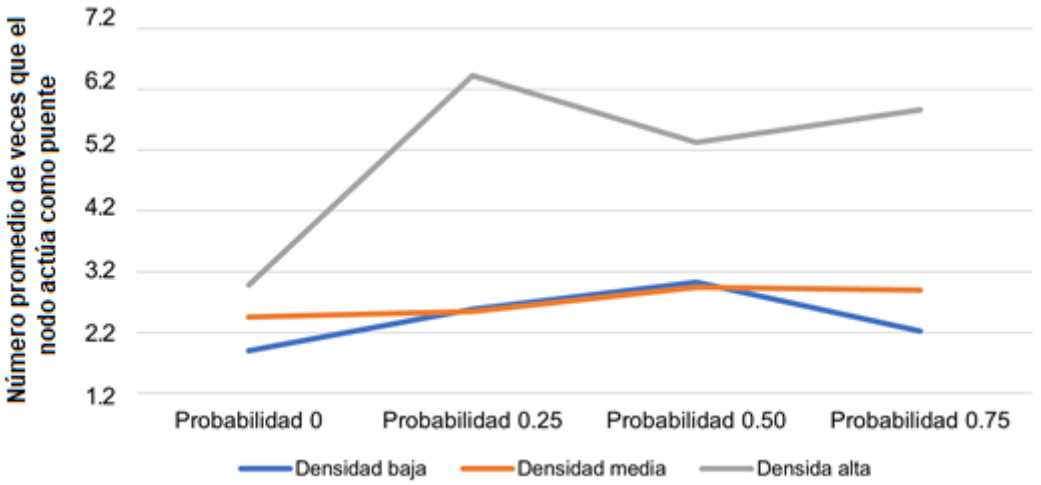

Figura 3.18. Intermediación para intervalo de 5 minutos

Cuando se configura un intervalo de reenrutamiento de 5 minutos, los valores obtenidos para las densidades baja y media son muy similares y la intermediación es baja, lo que puede interpretarse como redes en las que se forman pocas conexiones, debido a que no existe tanta congestión vehicular. Para la densidad alta el mejor valor se presenta nuevamente con la probabilidad de 50%, aunque este valor está muy distante respecto a las curvas con las otras densidades.

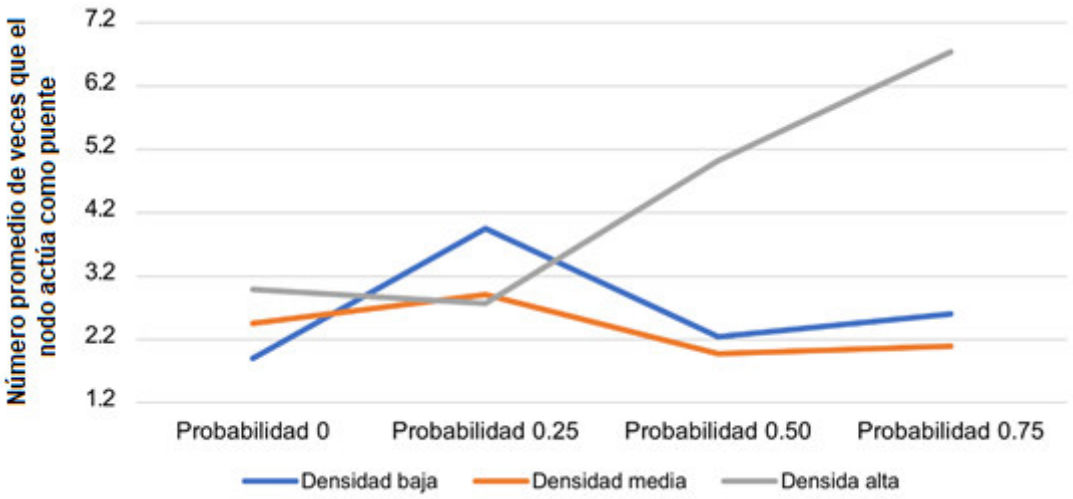

Figura 3.19. Intermediación para intervalo de 10 minutos

Por último, con el intervalo de 10 minutos se puede notar que independientemente de la densidad vehicular, la intermediación empeora respecto a los valores obtenidos para un escenario sin reenrutamiento (probabilidad 0%), ya que, aunque varíe la cantidad de vehículos que buscan nuevas rutas, el intervalo con el que lo hacen no es suficiente para mejor el rendimiento de la red.

3.4. DISTANCIA MEDIA

La distancia media corresponde a la cantidad promedio de saltos que se deben dar para conectar dos nodos. Esta métrica está muy relacionada con el tamaño de la red y la cantidad de conexiones que se formen entre los nodos.

Para un primer análisis de esta métrica se establece una probabilidad fija y se varían los intervalos de reenrutamiento para cada densidad vehicular.

En el caso de la probabilidad de 25% que se presenta en la Figura 3.20, se puede observar que los mejores valores de la densidad media se obtienen en los intervalos de 3 y 5 minutos, lo que significa que en promedio las rutas son más cortas para estos casos.

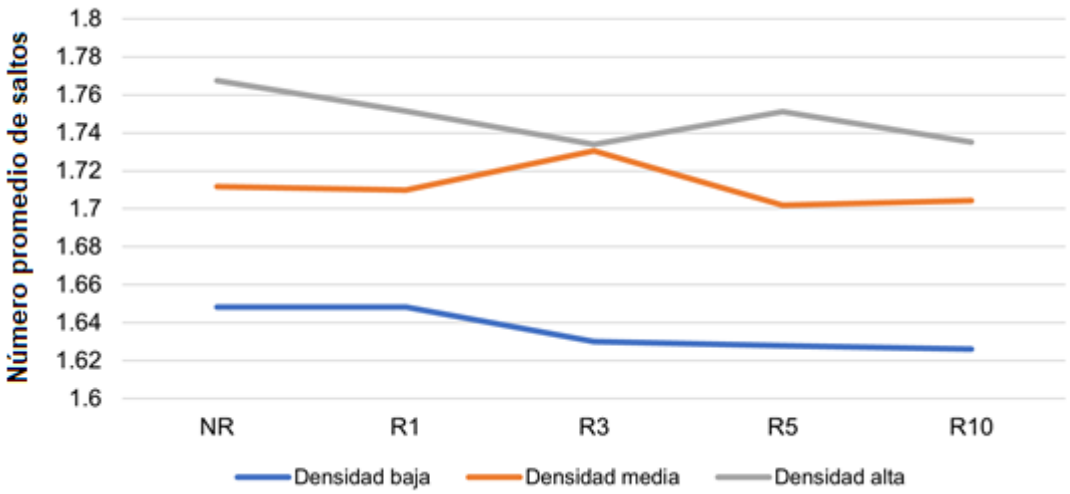

Figura 3.20. Distancia media para probabilidad de 25%

Para las probabilidades de 50% y 75% el comportamiento es bastante parecido, pues los escenarios con mayor cantidad de vehículos (densidad media y alta) presentan una valor más alto para la distancia media. A partir de esto se puede inferir que la distancia media está directamente relacionada con el número de nodos presentes en la red, puesto que a medida que aumentan los nodos, también se incrementa el número de saltos que se deben dar para conectar dos nodos dentro de la red.

Con estas dos probabilidades, también se puede comprobar que los valores mejoran cuando se utilizan los intervalos de reenrutamiento de 1, 3 y 5 minutos.

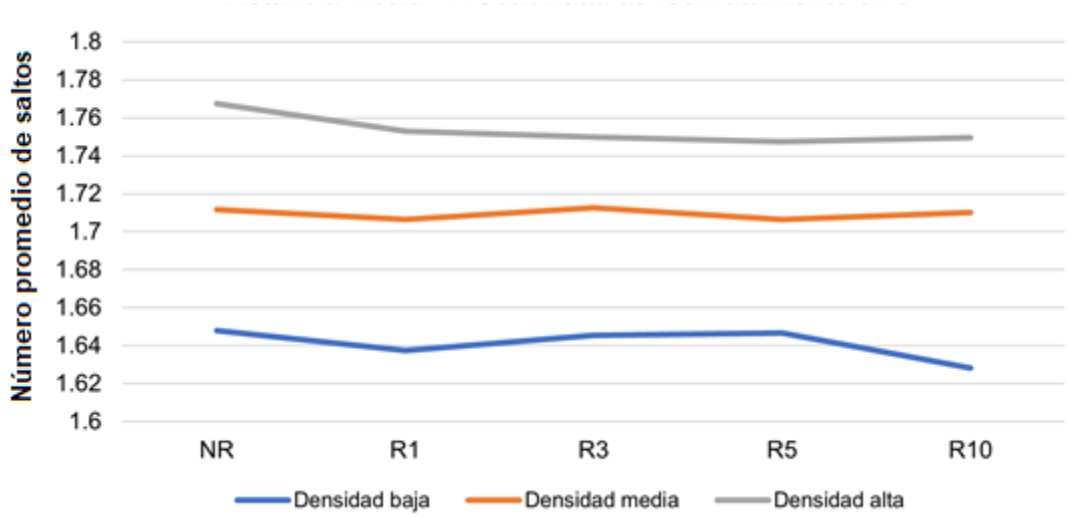

Figura 3.21. Distancia media para probabilidad de 50%

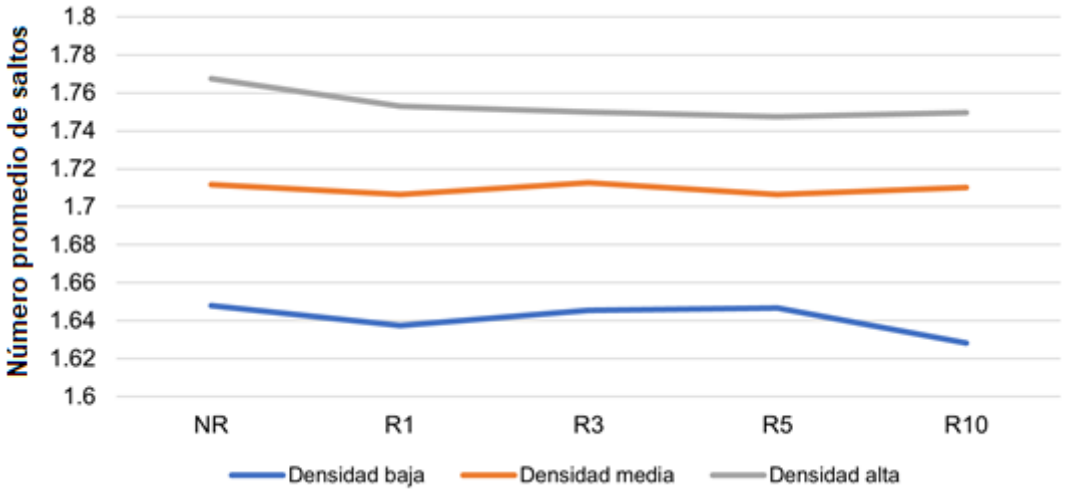

Figura 3.22. Distancia media para probabilidad de 75%

Para un segundo análisis de esta métrica se toma fijos los intervalos de 3 y 5 minutos y se varía la densidad y probabilidad de reenrutamiento, considerando que estos intervalos son de especial interés, ya que han presentado los mejores valores para las métricas anteriores.

Cuando el intervalo es 3 minutos, las probabilidades de 25% y 75% tienen los valores más bajos para las densidades baja y media, mientras que, para la densidad más alta, la probabilidad de 50% representa una menor distancia media. Esto sugiere que, dependiendo de la densidad vehicular, la mejor probabilidad puede variar para obtener un rendimiento óptimo.

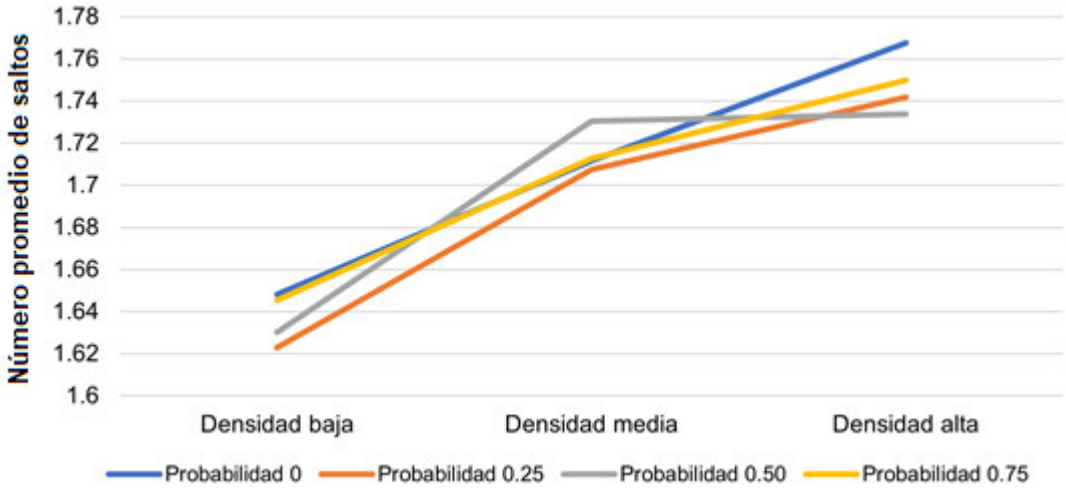

Figura 3.23. Distancia media para intervalo de 3 minutos

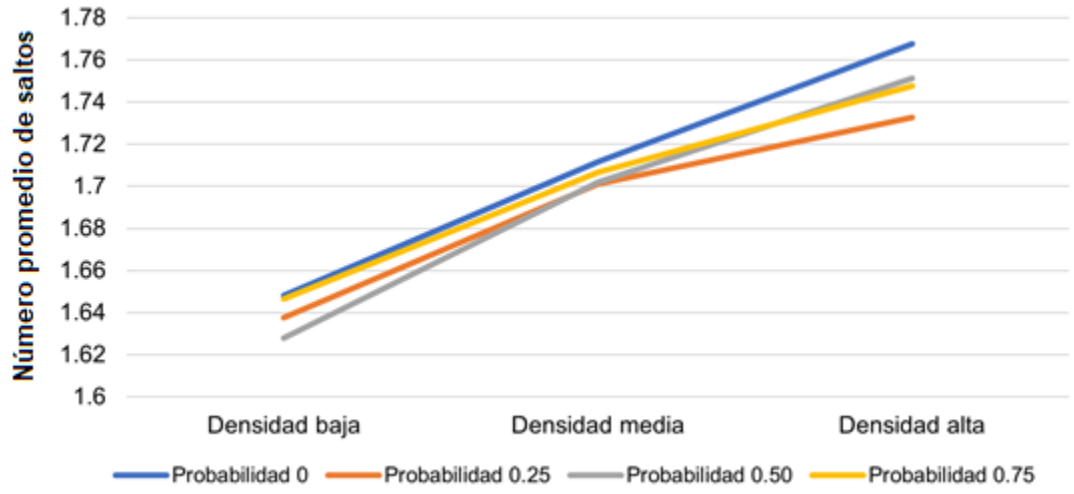

Figura 3.24. Distancia media para intervalo de 5 minutos

3.5. DIÁMETRO

Otra métrica que está directamente relacionada con el tamaño de la red es el diámetro, pues representa la ruta que conecta a los nodos más alejados de la red. En la Figura 3.25 se presentan los valores del diámetro para un intervalo de reenrutamiento de 3 minutos y cada curva representa cada densidad vehicular.

Para este caso en concreto, en promedio se necesita hasta un salto adicional en los escenarios con densidad media y alta respecto a los escenarios con menos vehículos (densidad baja).

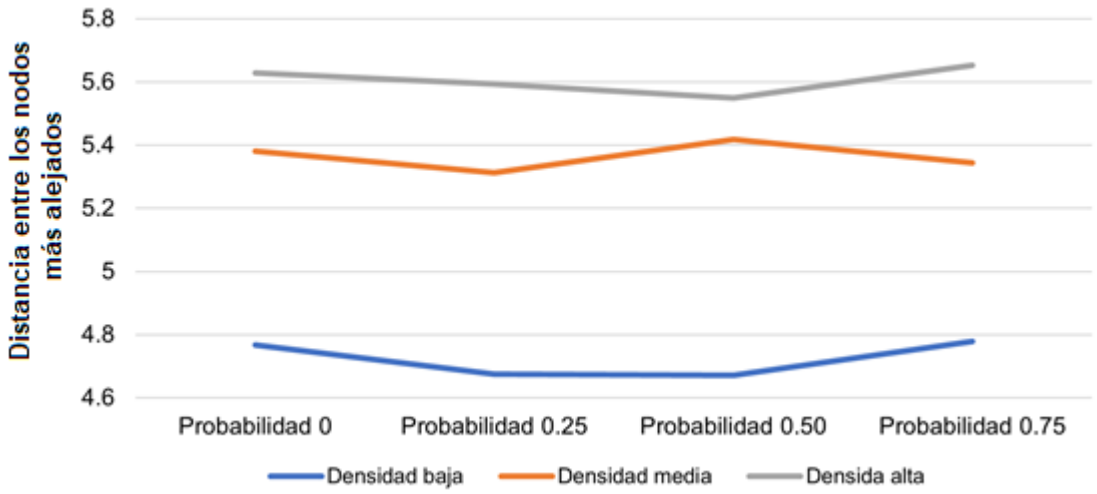

Figura 3.25. Diámetro promedio de cada instante para intervalo de 3 minutos

Cuando se establece un intervalo de reenrutamiento 5 minutos, los valores obtenidos son muy similares al caso anterior, puesto que, para la densidad alta, en promedio se necesitan 5.6 saltos para conectar los nodos más alejados, mientras que, en la densidad baja, este valor se reduce a 4.8 aproximadamente

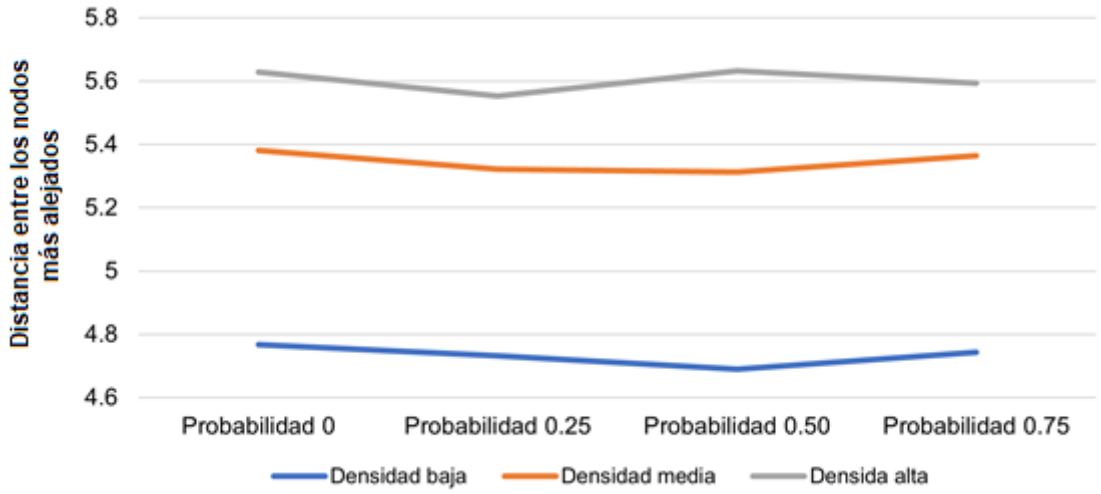

Figura 3.26. Diámetro para intervalo de 5 minutos

Una segunda manera de estudiar esta métrica es tomar un valor fijo de probabilidad (en este caso 50% que ha presentado un mejor rendimiento para las métricas anteriores) y se dibujan las curvas para cada intervalo de reenrutamiento. Aquí se puede notar que el diámetro está directamente relacionado con el número de nodos presentes en la red, ya que, este crece a medida que se incrementa la cantidad de vehículos en la simulación.

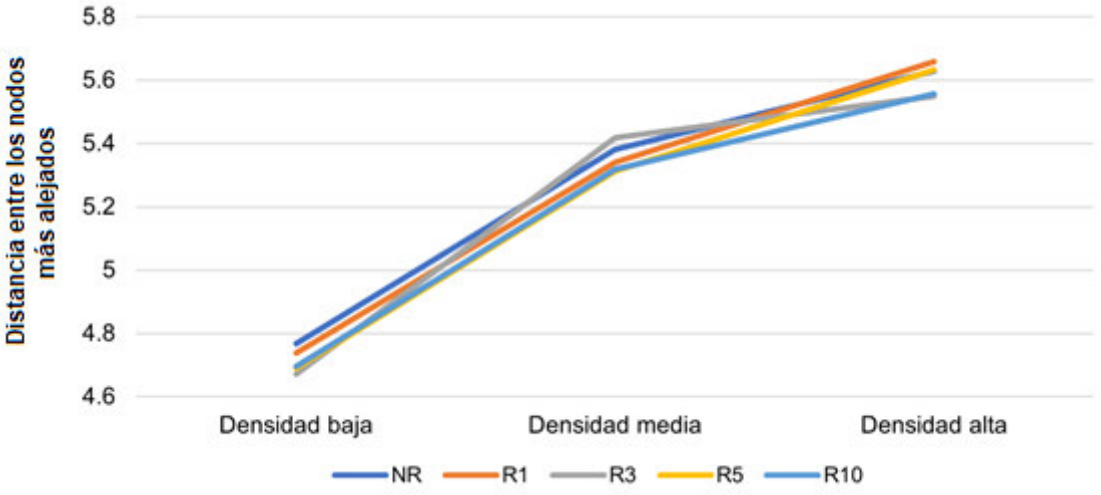

Figura 3.27. Diámetro para probabilidad de 50%

En base a los resultados obtenidos, un valor alto significa que se necesitan muchos saltos para conectar estos dos nodos. En términos de las comunicaciones vehiculares, un

diámetro alto sugiere la posibilidad de utilizar un protocolo de enrutamiento sobre esta red, es decir, los escenarios de densidad vehicular alta pueden prestarse a la aplicación de estos protocolos.

3.6. DENSIDAD DE ARISTAS

La densidad de aristas es la relación entre los pares de nodos conectados en la red respecto a todos los posibles. Un valor bajo (cercano a 0) es muestra de poca conectividad entre los nodos dentro de la red, mientras que un valor alto (cercano a 1) indica conectividad total entre los nodos que componen el grafo.

En la Figura 3.2 se muestra la densidad de aristas obtenida en función de los valores promedio de todos los instantes cuando la probabilidad es igual al 50%. Se puede observar que el comportamiento para los distintos períodos de reenrutamiento es bastante similar, a medida que aumenta la cantidad de autos en la red, la densidad de aristas disminuye.

Si se comparan las curvas cuando se permite el reenrutamiento con la curva del escenario original, se nota que los valores disminuyen, principalmente porque cuando se permite el cambio de ruta, los vehículos pueden modificar sus rutas planificadas para evitar calles congestionadas y encontrar menos vehículos en las nuevas rutas. Este mismo comportamiento se presenta cuando se fija una probabilidad igual al 75%, tal como se muestra en la Figura 3.29.

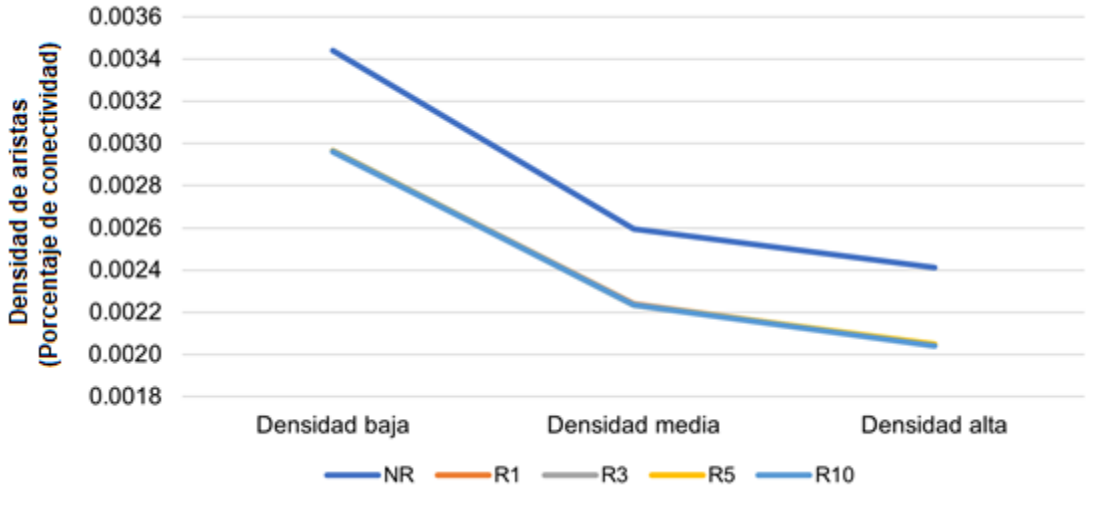

Figura 3.28. Densidad de aristas para probabilidad de 50% (promedio)

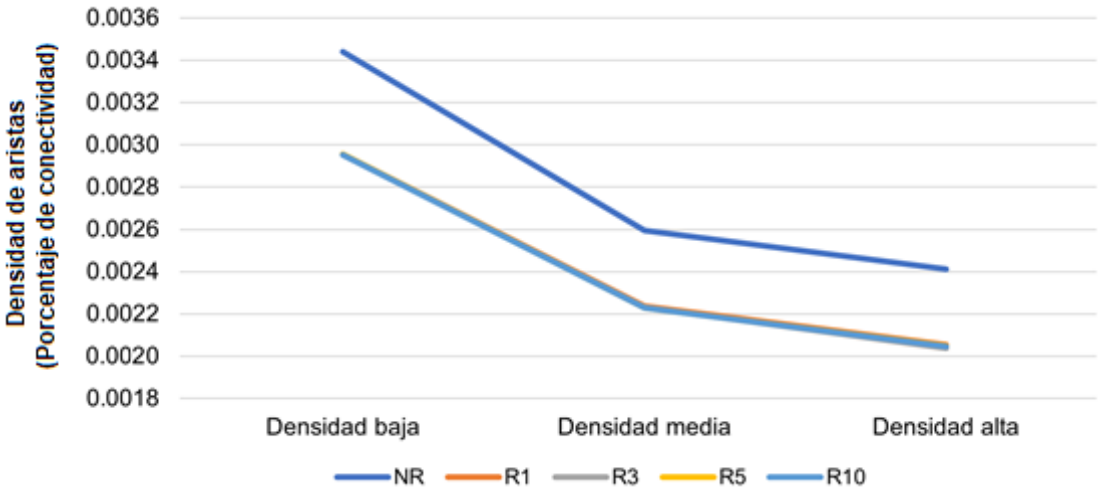

Figura 3.29. Densidad de aristas para probabilidad de 75% (promedio)

Cuando se obtienen los valores en función del grafo resumen, se puede observar que cuando la densidad de vehículos es baja, la densidad de aristas se ve afectada de forma negativa, debido a que los vehículos calculan nuevas rutas hacia sus destinos para evitar el tráfico y por esto se encuentran alejados y no forman muchas conexiones con sus vecinos.

Por el otro lado, cuando se incrementa la cantidad de vehículos en la simulación, crece también la densidad de aristas, ya que los vehículos están calculando nuevas rutas más cortas, pero se encuentran con mayor congestión, por lo que el número de conexiones formadas también se incrementa.

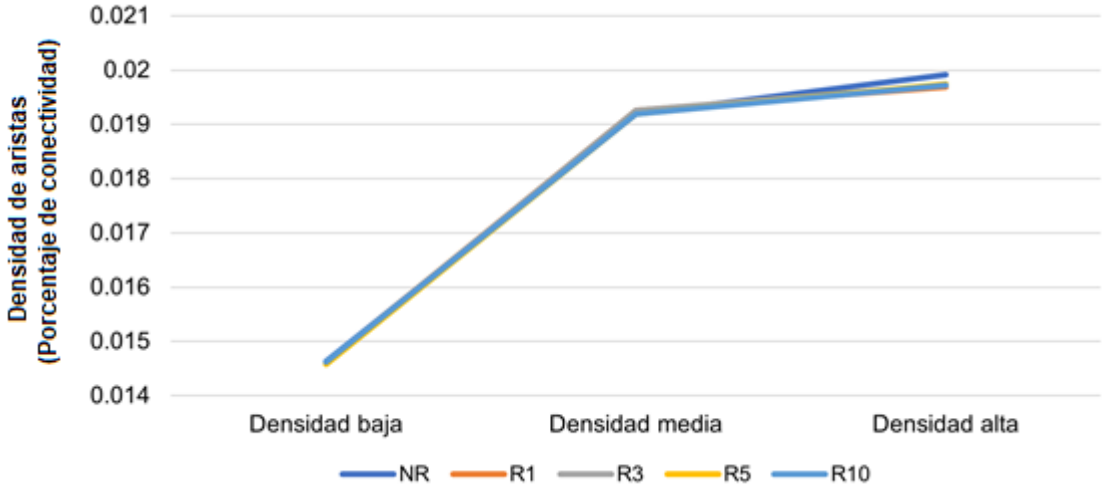

Figura 3.30. Densidad de aristas para probabilidad de 50% (resumen)

En el caso de la probabilidad de 75%, el comportamiento es idéntico al caso anterior, aunque en densidad baja se observan ligeras diferencias entre los diferentes intervalos de reenrutamiento, siendo el período de 3 minutos el que presenta el valor más bajo entre todos.

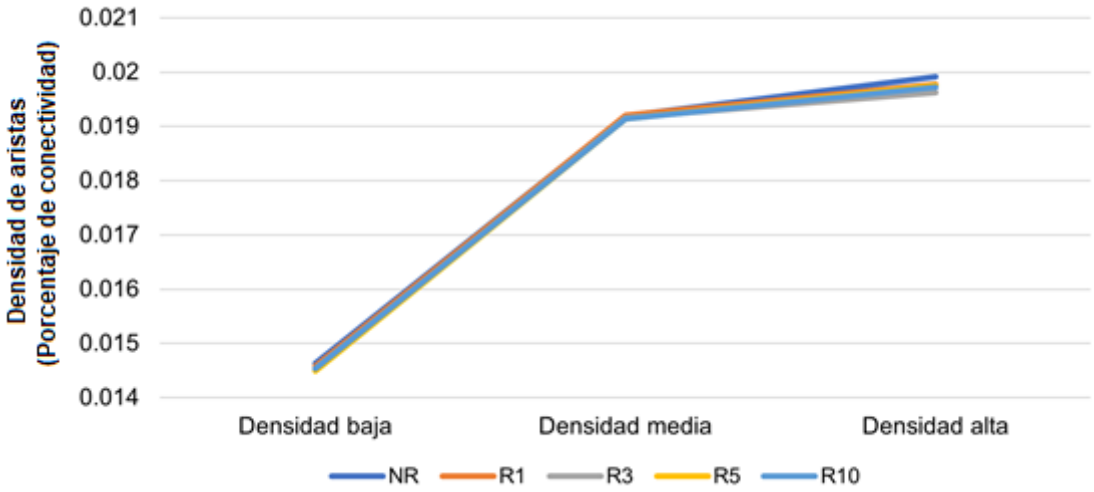

Figura 3.31. Densidad de aristas para probabilidad de 75% (resumen)

A partir de este comportamiento, se puede deducir que en los escenarios con una densidad de vehículos baja, la habilitación de la capacidad de reenrutamiento afecta negativamente en la conectividad de los nodos dentro de la red.

3.7. TRANSITIVIDAD

La transitividad puede tomarse como la proporción de nodos que tienen nodos adyacentes interconectados, y está muy relacionada con la cantidad de nodos presentes en la red. Un primer análisis de esta red se realiza con un valor de probabilidad de 25% y se dibujan las curvas para cada intervalo de reenrutamiento.

El comportamiento es muy similar a la densidad de aristas, en las que el valor de la transitividad es bajo existen pocos vehículos en la red, pues esos vehículos calculan nuevas rutas con menos congestión, se forman menos conexiones entre estos nodos y por lo tanto disminuye la probabilidad que estos formen tríadas. Comparado con los escenarios sin reenrutamiento (NR), los escenarios que permiten los cambios de rutas presentan una menor transitividad. Esto significa una mejora en términos de la congestión vehicular, pero resulta negativo desde el punto de vista de la conectividad entre los vehículos.

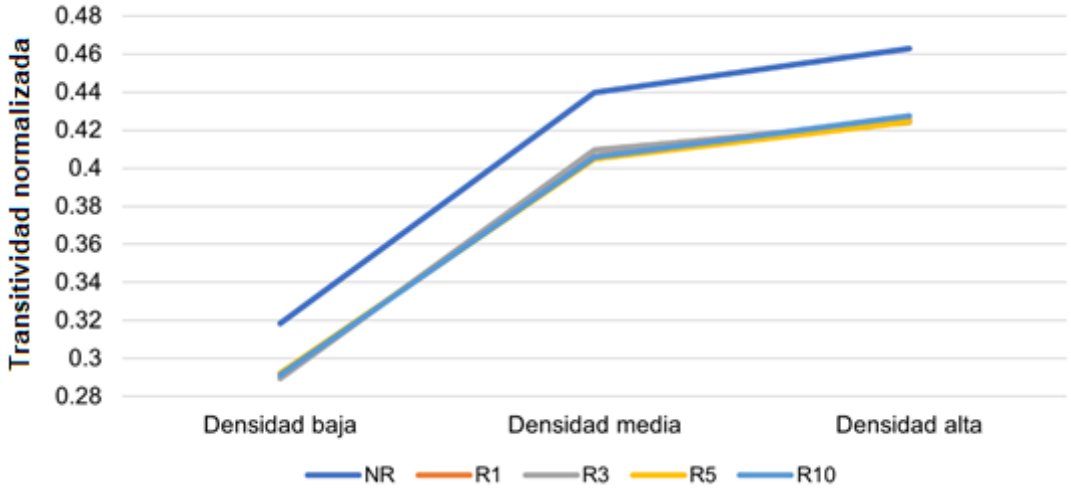

Figura 3.32. Transitividad para probabilidad de 25%

Se pueden utilizan los mismos datos para dibujar las curvas para cada caso de densidad vehicular y observar cómo se comportan los distintos intervalos de reenrutamiento. Como se indicó en la figura anterior, la transitividad disminuye cuando se habilita la capacidad de reenrutamiento respecto al escenario original.

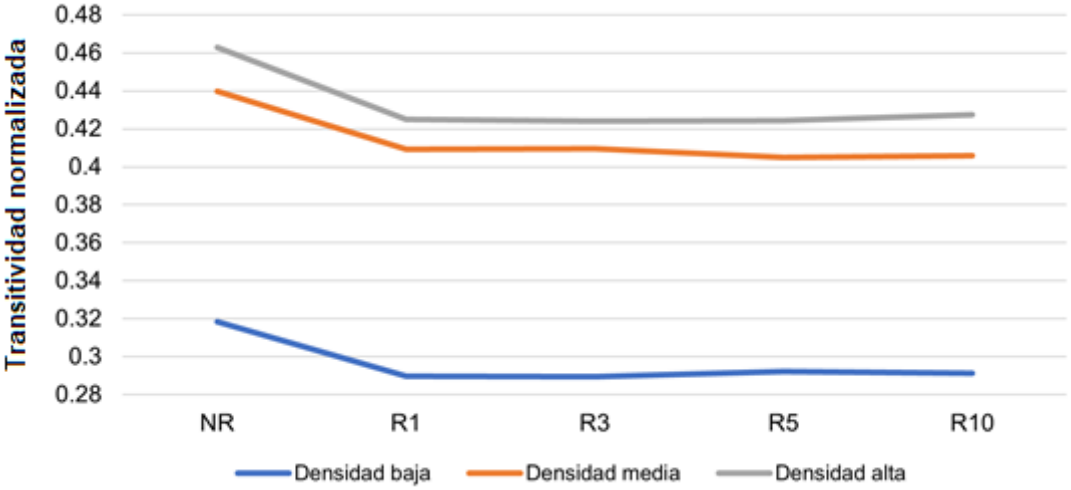

Figura 3.33. Transitividad para probabilidad de 25%

En la Figura 3.34, se muestra este mismo análisis tomando un valor de probabilidad igual a 75%. Se puede notar que, para las densidades media y alta, cuando se elige un período de 3 minutos, se obtienen los mejores resultados en términos de la congestión, pues el valor de transitividad es más bajo, lo que se traduce en menos vehículos que encuentran tráfico durante su ruta.

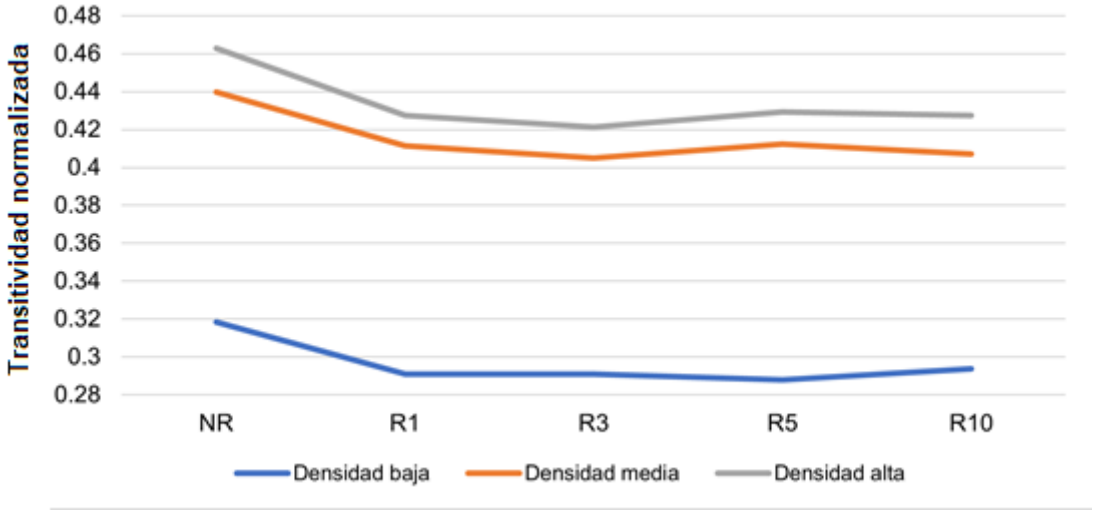

Figura 3.34. Transitividad para probabilidad de 75%

3.8. NÚMERO DE COMUNIDADES

Por último, se analiza el número de comunidades formadas para inferir como se podrían aplicar protocolos de enrutamiento basados en clúster en las redes formadas por los vehículos. Para esto se fija un valor de probabilidad y se dibujan las curvas para densidad de autos. Cuando la probabilidad es igual a 25% se observa que en los escenarios con pocos vehículos el número de comunidades formadas aumenta cuando se permite el reenrutamiento, especialmente cuando se calculan nuevas rutas cada 3 o 5 minutos, mientras que cuando aumenta la densidad, estos valores son más estables para los distintos intervalos.

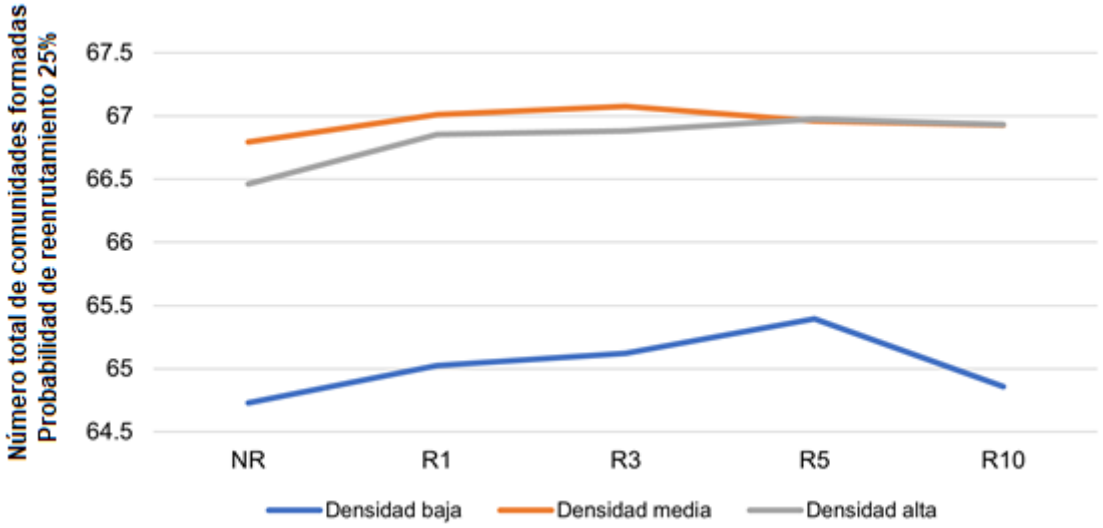

Figura 3.35. Número de comunidades para probabilidad de 25%

Este mismo comportamiento se presenta cuando se analizan las probabilidades de 50% y 75%, aunque para los casos de densidad media y alta, los mejores valores se obtienen cuando se selecciona un período de reenrutamiento de 5 minutos.

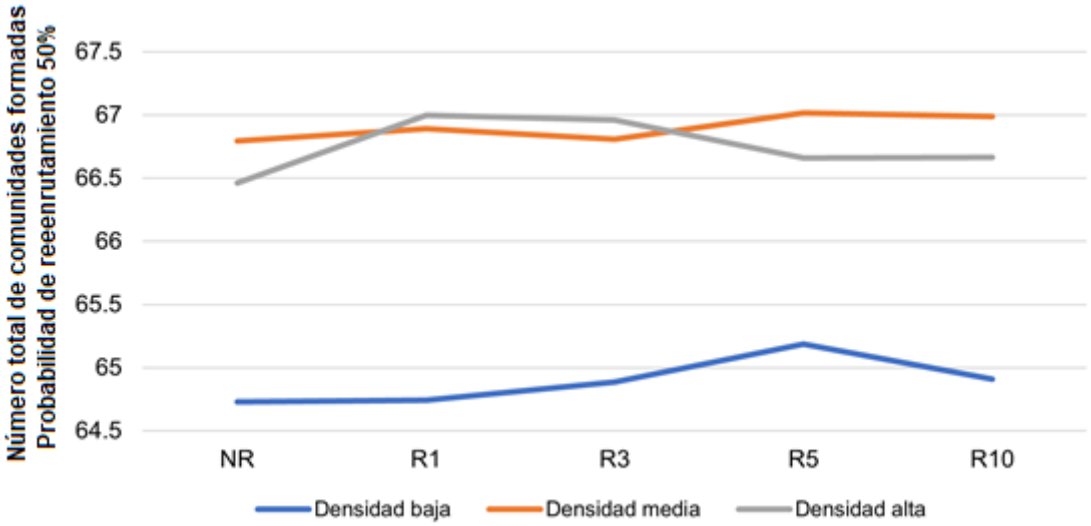

Figura 3.36. Número comunidades para probabilidad de 50%

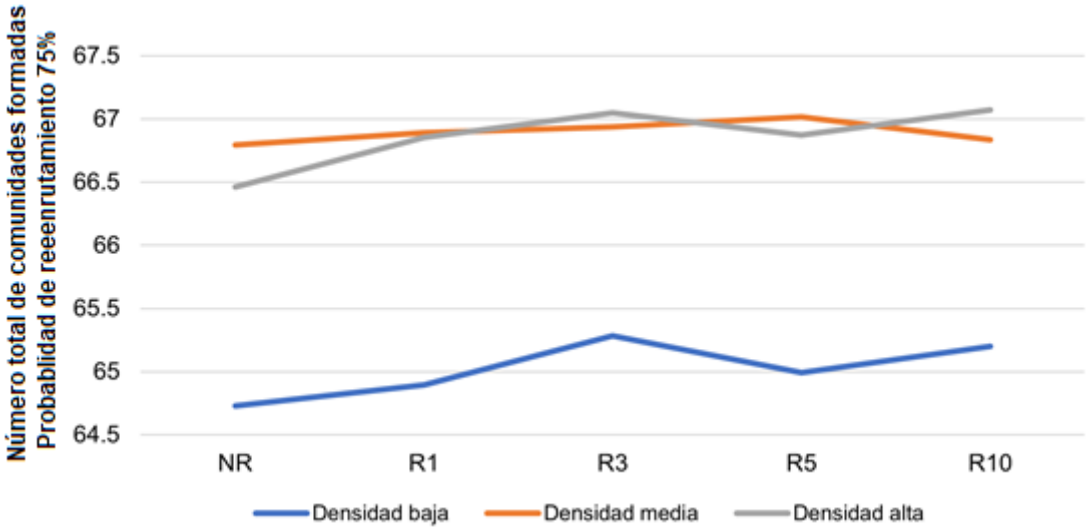

Figura 3.37. Número de comunidades para probabilidad de 75%

En base a los resultados obtenidos, se puede decir que el número de comunidades, al igual que el resto de las métricas analizadas, está muy relacionado con el número de nodos en la red, ya que, si existen más vehículos dentro de la red, la probabilidad que estos se conecten entre sí es mayor y por lo tanto el número de comunidades que se forman entre ellos también aumenta significativamente.

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

Las conclusiones que se derivan de la realización de este trabajo de titulación son las siguientes:

- Cuando se trabaja con sistemas de movilidad y tráfico vehicular, las simulaciones son el primer enfoque o alternativa que debería ser utilizado los investigadores para poder evaluar el rendimiento de estas redes vehiculares, porque la implementación de un escenario real resulta muy complejo y costoso.
- Con la habilitación de la capacidad de reenrutamiento en las simulaciones, se modifica el comportamiento de los vehículos presentes en la red, minimizando los niveles de congestión y permitiendo que los vehículos se encuentren más dispersos en la red.
- Al configurar el reenrutamiento en SUMO, se evitan situaciones no realistas como el tráfico en escenarios con poca cantidad de vehículos, que si se producen cuando se utilizan las rutas estáticas en la configuración por defecto del simulador.
- El reenrutamiento permite obtener escenarios de simulación más realistas, ya que en la actualidad muchos conductores disponen de dispositivos que les permiten buscar rutas más cortas hacia sus destinos y evitar así las calles que presentan congestión. Con esta característica de SUMO se puede emular este comportamiento.
- Los valores de las métricas están muy relacionados con el número de nodos presenten en la red, a medida que se incrementa la cantidad de vehículos, aumenta el número de conexiones formadas y con esto, se incrementan los valores obtenidos para las métricas.
- Desde el punto de vista de las métricas analizadas, la característica de reenrutamiento tiene un gran impacto en la conectividad de los vehículos, ya que estos empiezan a transitar por rutas menos congestionadas y por lo tanto se forman menos conexiones directas entre estos nodos, en especial en los escenarios con una baja densidad de autos.
- La teoría de grafos y en especial las métricas de centralidad son una herramienta muy útil el análisis redes vehiculares en las que se disponen de miles de nodos y representan una forma más visual de interpretar los reportes obtenidos en las simulaciones.

- Los lenguajes de programación AWK y R son indicados para el análisis y procesamiento de grandes cantidades de información, por la facilidad en la sintaxis, la curva de aprendizaje y la gran cantidad de herramientas para el trabajo con grafos y su manipulación.

4.2. RECOMENDACIONES

Las recomendaciones relacionadas con el desarrollo de este trabajo de titulación son las siguientes:

- Elegir un intervalo de reenrutamiento no tan bajo, ya que es una condición poco realista porque los conductores no buscan nuevas rutas hacia sus destinos con tanta frecuencia. Tampoco debe se debe seleccionar un valor muy alto, ya que esto significa un comportamiento prácticamente similar al uso de las rutas estáticas.

- Seleccionar un valor de probabilidad adecuado dependiendo de la densidad vehicular, pues al elegir valores muy bajos el comportamiento es similar a no habilitar el reenrutamiento y seguir utilizando las rutas estáticas y con un valor demasiado alto, se simula una situación no realista en la que la mayoría de vehículos calculan nuevas rutas hacia sus destinos, lo cual afecta a los resultados obtenidos y sobre todo no representa una situación que se presente en la vida real.

- Utilizar la versión más reciente de SUMO, ya que esta contiene actualizaciones en las herramientas para la importación de los mapas, la depuración de este escenario, la generación de la demanda de tráfico y nuevas características que permiten optimizar los tiempos que toma la ejecución de una simulación.

- Antes de ejecutar los programas para el procesamiento de los reportes, instalar el software necesario para soportar los comandos AWK y configurar en las variables de entorno del sistema las rutas para que el sistema operativo reconozca desde cualquier directorio estos comandos. Para este trabajo se utilizó Windows con la herramienta GWAK.

- Realizar las simulaciones y el procesamiento de los datos en un ordenador que disponga de buenas prestaciones, teniendo especial cuidado en la memoria RAM y el procesador, ya que la demanda de recursos computaciones para ejecutar las simulaciones con mucha información y el procesamiento de los reportes obtenidos que contienen millones de líneas es significativamente alta.

- Organizar los archivos y reportes de cada escenario de simulación en un directorio separado, puesto que durante el procesamiento se necesita acceder a cada uno de ellos y además se genera una gran cantidad de archivos nuevos y para los nodos y aristas, que pueden confundirse e incluso ser sobrescritos si se trabaja en un único directorio para todos los casos de simulación.

4.3. TRABAJOS FUTUROS

Con los resultados obtenidos en este trabajo de titulación junto con los datos recogidos en proyectos similares se pueden proponer los siguientes temas de investigación que pueden realizarse en un futuro:

- Utilizar diferentes herramientas para la generación de demanda de tráfico y comparar los efectos del reenrutamiento sobre cada una.
- Configurar características de SUMO para la simulación de tráfico peatonal, simulación de accidentes o zonas bloqueadas dentro de la red y medir el impacto del reenrutamiento en estas circunstancias.
- Modificar los programas de procesamiento en R para utilizar las características de paralelización y reducir los tiempos de ejecución de estos programas.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] "Congestión de Tránsito, el problema y cómo enfrentarlo", p. 196.

[2] I. N. de E. y Censos, "Anuario de Estadística de Transporte", *Instituto Nacional de Estadística y Censos*. [Online]. Disponible en: https://www.ecuadorencifras.gob.ec/documentos/web-inec/Estadisticas_Economicas/Estadistica%20de%20Transporte/2018/2018_ANET_BOLETIN.pdf. [Accedido: 02-ago-2020]

[3] M. C. V. Ortega, "Ciudades inteligentes y modelos logísticos de ciudad", p. 58.

[4] L. Codeca, R. Frank, S. Faye, y T. Engel, "Luxembourg SUMO Traffic (LuST) Scenario: Traffic Demand Evaluation", *IEEE Intell. Transp. Syst. Mag.*, vol. 9, nº 2, pp. 52–63, Summer 2017, doi: 10.1109/MITS.2017.2666585.

[5] P. B. Nec, "Los modelos de simulación: una herramienta multidisciplinar de investigación", p. 11.

[6] P. A. Lopez *et al.*, "Microscopic Traffic Simulation using SUMO", en *The 21st IEEE International Conference on Intelligent Transportation Systems*, 2018 [Online]. Disponible en: https://elib.dlr.de/124092/

[7] M. Behrisch, L. Bieker, J. Erdmann, y D. Krajzewicz, "SUMO – Simulation of Urban Mobility", p. 7.

[8] "SUMO - Simulation of Urban Mobility", *CIVITAS*. [Online]. Disponible en: http://civitas.eu/hr/tool-inventory/sumo-simulation-urban-mobility. [Accedido: 04-ene-2021]

[9] D. Krajzewicz, J. Erdmann, M. Behrisch, y L. Bieker, "Recent Development and Applications of SUMO – Simulation of Urban Mobility", p. 11, 2012.

[10] "Theory/Traffic Simulations - SUMO Documentation". [Online]. Disponible en: https://sumo.dlr.de/docs/Theory/Traffic_Simulations.html. [Accedido: 04-ene-2021]

[11] W. S. Coloma Gómez, "ANÁLISIS DE LAS HERRAMIENTAS DE GENERACIÓN DE DEMANDA DE TRÁFICO EN SUMO. CASO DE ESTUDIO: VÍAS DE ACCESO A QUITO.", Escuela Politécnica Nacional, Quito, 2019.

[12] P. Barbecho Bautista, L. Urquiza-Aguiar, y M. Aguilar Igartua, "Evaluation of Dynamic Route Planning Impact on Vehicular Communications with SUMO", en *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2020, pp. 27–35.

[13] “DUAROUTER”, *SUMO Documentation*. [Online]. Disponible en: https://sumo.dlr.de/docs/duarouter.html. [Accedido: 30-nov-2021]

[14] F. Alonso Sarría, “Introducción a AWK”. [Online]. Disponible en: https://www.um.es/innova/OCW/informatica-para-universitarios/ipu_docs/la_shell/awk.pdf

[15] “The GNU Awk User’s Guide”. [Online]. Disponible en: https://www.gnu.org/software/gawk/manual/gawk.html. [Accedido: 09-dic-2021]

[16] “AWK”, *Wikipedia, la enciclopedia libre*. 30-ago-2021 [Online]. Disponible en: https://es.wikipedia.org/w/index.php?title=AWK&oldid=138001996. [Accedido: 07-dic-2021]

[17] J. F. Yazlle, “Teoría de grafos”. Universidad Nacional de Salta [Online]. Disponible en: http://www.unsa.edu.ar/yazlle/public_html/discreta/grafos.pdf. [Accedido: 18-ene-2021]

[18] R. F. Capcha Ventura, “La teoría de grafos en la resolución de problemas aritméticos para estudiantes del Laboratorio de Investigación e Innovación Pedagógica de la Universidad Nacional Daniel Alcides Carrión de Pasco”, Universidad Nacional Daniel Alcides Carrión, Cerro de Pasco, México, 2015 [Online]. Disponible en: http://repositorio.undac.edu.pe/bitstream/undac/156/1/T026_45783407T.pdf

[19] J. Alonso y J. Carabalí, “Breve tutorial para visualizar y calcular métricas de Redes (grafos) en R (para Economistas)”, vol. 7, 2019.

[20] O. C. García, “Redes y Sistemas Complejos Cuarto Curso del Grado en Ingeniería Informática”, p. 92.

[21] L. Urquiza-Aguiar, W. Coloma-Gomez, P. B. Bautista, y X. Calderon-Hinojosa, “Comparison of SUMO’s vehicular demand generators in vehicular communications via graph-theory metrics$”, p. 13.

[22] J. A. Ahumada, “R para Principiantes”, *Univ. Hawaii*, 2003.

[23] J. S. Santana y E. M. Farfán, “El arte de programar en R: un lenguaje para la estadística”, p. 197.

[24] “netconvert - SUMO Documentation”. [Online]. Disponible en: https://sumo.dlr.de/docs/netconvert.html. [Accedido: 10-dic-2021]

[25] “netedit - SUMO Documentation”. [Online]. Disponible en: https://sumo.dlr.de/docs/Netedit/index.html. [Accedido: 10-dic-2021]

[26] “Intracantonal Urbano”, *Secretaría de Movilidad de Quito*. [Online]. Disponible en: https://secretariademovilidad.quito.gob.ec/index.php/intracantonal-urbano. [Accedido: 10-dic-2021]

[27] C. Sommer, D. Eckhoff, R. German, y F. Dressler, “A computationally inexpensive empirical model of IEEE 802.11p radio shadowing in urban environments”, en *2011 Eighth International Conference on Wireless On-Demand Network Systems and Services*, 2011, pp. 84–90, doi: 10.1109/WONS.2011.5720204.

ANEXOS

ANEXO A. Estadísticas de tráfico proporcionadas por la Secretaría de Movilidad del Distrito Metropolitano de Quito

ANEXO B. Tiempos de simulación de los casos propuestos

ANEXO C. Código del programa Get Data

ANEXO D. Código de la función Get Unit Info

ANEXO E. Código del programa maestro

ANEXO F. Organización de los archivos

ANEXO A

ESTADÍSTICAS DE TRÁFICO PROPORCIONADAS POR LA SECRETARÍA DE MOVILIDAD DEL DISTRITO METROPOLITANO DE QUITO (6 AM – 1 PM)

CRUCE	SENTIDO	Entrada / Salida	06	07	08	09	10	11	12	13
Av. 6 de Diciembre y Orellana	NORTE-SUR	Salida	529	909	1184	1126	1088	1127	1091	1076
Av. 6 de Diciembre y Orellana	SUR-NORTE	Entrada	512	918	1009	875	930	1031	1069	1125
Av. 6 de Diciembre y Orellana	OCCIDENTE-ORIENTE	Entrada	587	1028	1414	1373	1402	1474	1566	1589
Av. 6 de Diciembre y Orellana	ORIENTE-OCCIDENTE	Salida	1188	1349	1483	1357	1402	1399	1375	1356
Orellana y Diego de Almagro	SUR-NORTE	Entrada	102	200	302	300	305	337	361	358
Av. Orellana y Juan Leon Mera	SUR-NORTE	Entrada	200	448	602	529	544	596	630	599
Av. Amazonas y Cristóbal Colón	NORTE-SUR	Salida	603	827	827	913	938	944	922	902
Av. Amazonas y Orellana	SUR-NORTE	Entrada	62	110	174	188	214	240	240	259
Av. 9 de Octubre y Orellana	SUR-NORTE	Entrada	241	553	650	616	638	669	660	676
Av. 10 de Agosto y Cristóbal Colón	NORTE-SUR	Salida	1245	1760	1699	1735	1737	1771	1752	1704
Av. 10 de Agosto y Cristóbal Colón	SUR-NORTE	Entrada	1088	1579	1516	1369	1410	1426	1614	1540
Av. 6 de Diciembre y Granados	OCCIDENTE-ORIENTE	Salida	123	223	221	208	234	261	270	263
Av. 6 de Diciembre y Granados	ORIENTE-OCCIDENTE	Entrada	634	1108	1134	1183	1206	1181	1184	1162
Av. Eloy Alfaro y Gaspar De Villarroel Entrada	NORTE-SUR	Entrada	1088	1454	1604	1418	1423	1389	1403	1303
Av. Eloy Alfaro y Av. de Los Granados	SUR-NORTE	Salida	660	1207	1285	1253	1191	1289	1404	1445
Av. Eloy Alfaro y Gaspar De Villarroel Urrutia	OCCIDENTE-ORIENTE	Salida	266	407	433	485	505	563	614	625
Av. Eloy Alfaro y Gaspar De Villarroel Urrutia	ORIENTE-OCCIDENTE	Salida	219	212	211	215	214	216	216	217
Av. 6 de Diciembre y Berlanga PP1	NORTE-SUR	Entrada	687	905	941	976	996	992	946	897
Av. 6 de Diciembre y Berlanga PP2	SUR-NORTE	Salida	305	515	594	622	600	654	694	688
Av. De los Shyris y Tomas de Berlanga	NORTE-SUR	Entrada	838	1280	1397	1090	1088	1090	1094	1073
Av. De los Shyris y Tomas de Berlanga	SUR-NORTE	Salida	434	828	1051	1051	1047	1146	1289	1325
Av. Amazonas y Tomas de Berlanga	NORTE-SUR	Entrada	1084	1546	1503	1433	1489	1451	1345	1247
Av. Amazonas y Gaspar de Villarroel	SUR-NORTE	Salida	466	1004	1385	1334	1315	1427	1555	1545
Av. 10 de Agosto y Juan de Ascaray	GIRO DERECHA SUR-NORTE	Salida	290	574	725	698	761	845	884	867
Av. 10 de Agosto y Juan de Ascaray	SUR-NORTE	Salida	606	1153	1377	1450	1507	1606	1728	1705
Av. 10 de Agosto y Juan de Ascaray	ORIENTE-OCCIDENTE	Entrada	374	613	825	803	766	770	749	742
Av. 10 de Agosto y Falconi	NORTE-SUR	Entrada	943	1192	1253	1037	1104	1076	1003	914
Av. 10 de Agosto y Falconi	SUR-NORTE	Salida	461	796	883	895	832	881	954	963
Av. 10 de Agosto y Falconi	SUR-NORTE	Salida	600	983	1125	956	931	1022	1090	1106

Av. La Prensa y El Inca	SUR-NORTE	Salida	491	921	1135	1219	1220	1309	1381	1375
Av. La Prensa y Fray Jose Falconi	NORTE-SUR	Entrada	1078	1721	1805	1419	1449	1424	1383	1306
Av. Brasil y Echeverria	NORTE-SUR	Entrada	340	686	818	705	772	741	704	680
Av. Brasil y Echeverria	SUR-NORTE	Salida	266	552	698	790	840	918	1053	1062
Av. America y Mariana de Jesus	ORIENTE-OCCIDENTE	Salida	408	600	748	732	722	730	737	751
Ulloa y Mariana de Jesus	OCCIDENTE-ORIENTE	Entrada	434	696	719	786	821	809	810	821
Ulloa y Atahualpa	OCCIDENTE-ORIENTE	Entrada	512	978	1174	796	746	752	800	781
Ulloa y Atahualpa	ORIENTE-OCCIDENTE	Salida	662	950	1223	1183	1189	1210	1204	1246
Av. America y Rumipamba	NORTE-SUR	Salida	728	917	1079	1167	1243	1332	1320	1227
Av. America y Rumipamba	SUR-NORTE	Entrada	637	1053	1134	1068	1092	1161	1168	1125
Av. America y Republica NN. UU	GIRO DERECHA SUR-NORTE	Salida	261	650	801	799	854	900	992	1001
Av. America y Republica NN. UU	GIRO DERECHA SUR-NORTE	Salida	316	752	874	867	916	958	1068	1085
Av. 10 de Agosto y Naciones Unidas	ORIENTE-OCCIDENTE	Salida	351	633	791	841	835	877	912	939
Av. República y Mañosca	NORTE-SUR	Entrada	608	919	1123	963	945	926	902	836
Av. República y Mañosca	SUR-NORTE	Salida	748	1172	1226	1202	1253	1307	1433	1471
Av. Eloy Alfaro y Portugal	OCCIDENTE-ORIENTE	Salida	93	210	326	291	266	299	336	341
Av. Eloy Alfaro y Portugal	ORIENTE-OCCIDENTE	Entrada	73	129	192	182	182	187	204	212
Av. 6 de Diciembre y Republica	OCCIDENTE-ORIENTE	Salida	622	964	996	969	961	978	962	940
Av. 6 de Diciembre y Republica	ORIENTE-OCCIDENTE	Entrada	568	815	785	825	851	840	825	809
Av. 6 de Diciembre y Belgica - Interoceanica	ORIENTE-OCCIDENTE	Entrada	1221	2138	2389	1824	1637	1398	1292	1274
Av. Interoceanica y Gonzales Suarez	OCCIDENTE-ORIENTE	Salida	319	782	751	752	850	898	945	970
Av. Eloy Alfaro y Ayarza-El Batan	SUR-NORTE	Entrada	371	705	896	821	757	828	894	916
Av. Eloy Alfaro y Ayarza-El Batan	NORTE-SUR	Salida	357	575	723	634	614	644	681	662
Ulloa y Cuero y Caicedo	OCCIDENTE-ORIENTE	Entrada	181	312	373	296	290	448	388	289
Ulloa y Cuero y Caicedo	ORIENTE-OCCIDENTE	Salida	110	198	252	298	313	342	371	345
Av. America y Rumipamba	OCCIDENTE-ORIENTE	Entrada	222	255	291	239	231	212	227	227
Ulloa y Rumipamba	ORIENTE-OCCIDENTE	Salida	191	271	262	270	280	297	274	329
Av. America y Mañosca	OCCIDENTE-ORIENTE	Entrada	290	458	525	499	504	497	489	458
Av. America y Mañosca	ORIENTE-OCCIDENTE	Salida	137	270	301	307	299	324	351	359
		TOTAL	29030	46963	53217	50232	50749	52419	53808	53108

	Densidad alta
	Densidad media
	Densidad baja

ESTADÍSTICAS DE TRÁFICO PROPORCIONADAS POR LA SECRETARÍA DE MOVILIDAD DEL DISTRITO METROPOLITANO DE QUITO (2 PM – 10 PM)

CRUCE	SENTIDO	Entrada / Salida	14	15	16	17	18	19	20	21	22
Av. 6 de Diciembre y Orellana	NORTE-SUR	Salida	1019	1051	996	935	864	812	710	547	421
Av. 6 de Diciembre y Orellana	SUR-NORTE	Entrada	1027	1063	1081	1194	1087	886	690	500	356
Av. 6 de Diciembre y Orellana	OCCIDENTE-ORIENTE	Entrada	1463	1460	1417	1367	1327	1215	992	710	474
Av. 6 de Diciembre y Orellana	ORIENTE-OCCIDENTE	Salida	1322	1318	1303	1275	1093	979	846	636	418
Orellana y Diego de Almagro	SUR-NORTE	Entrada	355	360	347	361	295	261	202	207	112
Av. Orellana y Juan Leon Mera	SUR-NORTE	Entrada	601	600	580	540	537	460	363	255	188
Av. Amazonas y Cristóbal Colón	NORTE-SUR	Salida	933	927	909	858	880	904	805	652	461
Av. Amazonas y Orellana	SUR-NORTE	Entrada	208	211	206	303	202	121	92	73	50
Av. 9 de Octubre y Orellana	SUR-NORTE	Entrada	624	639	651	676	569	420	266	203	128
Av. 10 de Agosto y Cristóbal Colón	NORTE-SUR	Salida	1661	1655	1621	1663	1679	1616	1298	986	647
Av. 10 de Agosto y Cristóbal Colón	SUR-NORTE	Entrada	1504	1437	1448	1208	1313	1254	950	700	456
Av. 6 de Diciembre y Granados	OCCIDENTE-ORIENTE	Salida	258	264	251	253	269	273	196	125	66
Av. 6 de Diciembre y Granados	ORIENTE-OCCIDENTE	Entrada	1110	1108	1168	1109	1029	936	720	603	381
Av. Eloy Alfaro y Gaspar De Villarroel Entrada	NORTE-SUR	Entrada	1283	1376	1470	1500	1396	1329	968	686	466
Av. Eloy Alfaro y Av. de Los Granados	SUR-NORTE	Salida	1350	1373	1279	1358	1400	1253	1091	831	553
Av. Eloy Alfaro y Gaspar De Villarroel Urrutia	OCCIDENTE-ORIENTE	Salida	573	564	545	550	632	676	561	387	243
Av. Eloy Alfaro y Gaspar De Villarroel Urrutia	ORIENTE-OCCIDENTE	Salida	217	217	213	208	218	219	216	213	217
Av. 6 de Diciembre y Berlanga PP1	NORTE-SUR	Entrada	908	926	927	911	772	643	560	401	268
Av. 6 de Diciembre y Berlanga PP2	SUR-NORTE	Salida	679	660	678	681	684	677	577	379	215
Av. De los Shyris y Tomas de Berlanga	NORTE-SUR	Entrada	1046	1053	1019	975	855	763	592	386	227
Av. De los Shyris y Tomas de Berlanga	SUR-NORTE	Salida	1240	1207	1259	1416	1444	1286	1000	653	412
Av. Amazonas y Tomas de Berlanga	NORTE-SUR	Entrada	1356	1468	1438	1361	1171	996	882	720	516
Av. Amazonas y Gaspar de Villarroel	SUR-NORTE	Salida	1492	1474	1476	1559	1586	1422	1148	792	501
Av. 10 de Agosto y Juan de Ascaray	GIRO DERECHA SUR-NORTE	Salida	766	773	759	814	718	593	449	301	167
Av. 10 de Agosto y Juan de Ascaray	SUR-NORTE	Salida	1563	1568	1597	1719	1567	1494	1188	852	567
Av. 10 de Agosto y Juan de Ascaray	ORIENTE-OCCIDENTE	Entrada	716	745	727	755	642	509	395	294	186
Av. 10 de Agosto y Falconi	NORTE-SUR	Entrada	879	957	944	903	722	615	522	403	256
Av. 10 de Agosto y Falconi	SUR-NORTE	Salida	892	885	913	1035	1005	920	706	498	326
Av. 10 de Agosto y Falconi	SUR-NORTE	Salida	1060	1044	1045	1314	1306	1043	848	643	428
Av. La Prensa y El Inca	SUR-NORTE	Salida	1295	1283	1302	1361	1357	1431	1112	773	496
Av. La Prensa y Fray Jose Falconi	NORTE-SUR	Entrada	1276	1334	1307	1243	1057	904	780	594	398

Av. Brasil y Echeverria	NORTE-SUR	Entrada	695	743	687	674	393	92	53	17	9
Av. Brasil y Echeverria	SUR-NORTE	Salida	960	975	946	1108	1215	1084	828	574	368
Av. America y Mariana de Jesus	ORIENTE-OCCIDENTE	Salida	671	720	735	874	844	713	512	306	181
Ulloa y Mariana de Jesus	OCCIDENTE-ORIENTE	Entrada	786	795	777	763	701	661	506	347	209
Ulloa y Atahualpa	OCCIDENTE-ORIENTE	Entrada	731	753	748	770	647	555	409	273	148
Ulloa y Atahualpa	ORIENTE-OCCIDENTE	Salida	1072	1141	1156	1387	1177	902	660	400	229
Av. America y Rumipamba	NORTE-SUR	Salida	1092	1203	1183	1152	1024	997	830	646	398
Av. America y Rumipamba	SUR-NORTE	Entrada	1035	1052	1068	1089	1045	938	732	481	283
Av. America y Republica NN. UU	GIRO DERECHA SUR-NORTE	Salida	911	927	875	988	924	800	679	488	309
Av. America y Republica NN. UU	GIRO DERECHA SUR-NORTE	Salida	983	1009	968	1096	1033	876	731	534	328
Av. 10 de Agosto y Naciones Unidas	ORIENTE-OCCIDENTE	Salida	904	916	931	984	1028	958	779	557	358
Av. República y Mañosca	NORTE-SUR	Entrada	804	857	830	792	672	650	527	382	254
Av. República y Mañosca	SUR-NORTE	Salida	1501	1462	1399	1356	1271	1043	910	754	525
Av. Eloy Alfaro y Portugal	OCCIDENTE-ORIENTE	Salida	302	312	276	353	336	304	236	179	119
Av. Eloy Alfaro y Portugal	ORIENTE-OCCIDENTE	Entrada	192	186	171	191	184	160	116	82	51
Av. 6 de Diciembre y Republica	OCCIDENTE-ORIENTE	Salida	919	937	965	904	893	906	836	677	497
Av. 6 de Diciembre y Republica	ORIENTE-OCCIDENTE	Entrada	829	846	862	800	793	847	795	675	479
Av. 6 de Diciembre y Belgica - Interoceanica	ORIENTE-OCCIDENTE	Entrada	1211	1141	1016	1060	1007	917	743	522	395
Av. Interoceanica y Gonzales Suarez	OCCIDENTE-ORIENTE	Salida	926	937	967	951	965	908	737	524	332
Av. Eloy Alfaro y Ayarza-El Batan	SUR-NORTE	Entrada	798	783	722	803	758	661	585	420	313
Av. Eloy Alfaro y Ayarza-El Batan	NORTE-SUR	Salida	595	610	590	634	622	579	411	268	162
Ulloa y Cuero y Caicedo	OCCIDENTE-ORIENTE	Entrada	226	253	231	219	189	136	93	58	28
Ulloa y Cuero y Caicedo	ORIENTE-OCCIDENTE	Salida	319	307	282	334	303	175	120	83	42
Av. America y Rumipamba	OCCIDENTE-ORIENTE	Entrada	223	221	202	222	208	176	121	74	43
Ulloa y Rumipamba	ORIENTE-OCCIDENTE	Salida	241	251	250	272	236	198	148	77	33
Av. America y Mañosca	OCCIDENTE-ORIENTE	Entrada	464	481	471	444	390	374	328	243	147
Av. America y Mañosca	ORIENTE-OCCIDENTE	Salida	341	340	326	363	383	420	378	286	174
		TOTAL	50407	51158	50510	51988	48917	43940	35528	25930	17014

	Densidad alta
	Densidad media
	Densidad baja

ANEXO B

TIEMPOS DE EJECUCIÓN SIMULACIONES

DENSIDAD	PERÍODO	PROBABILIDAD	ESCENARIO	TIEMPO SIMULACIÓN
Baja		0.00	densidad-baja-NR	15154
	1	0.25	densidad-baja-R1-0.25	15155
		0.50	densidad-baja-R1-0.50	15229
		0.75	densidad-baja-R1-0.75	15152
	3	0.25	densidad-baja-R3-0.25	15156
		0.50	densidad-baja-R3-0.50	15154
		0.75	densidad-baja-R3-0.75	15155
	5	0.25	densidad-baja-R5-0.25	15281
		0.50	densidad-baja-R5-0.50	15155
		0.75	densidad-baja-R5-0.75	15155
	10	0.25	densidad-baja-R10-0.25	15229
		0.50	densidad-baja-R10-0.50	15230
		0.75	densidad-baja-R10-0.75	15154
Media		0.00	densidad-media-NR	15159
	1	0.25	densidad-media-R1-0.25	15160
		0.50	densidad-media-R1-0.50	15159
		0.75	densidad-media-R1-0.75	15161
	3	0.25	densidad-media-R3-0.25	15161
		0.50	densidad-media-R3-0.50	15160
		0.75	densidad-media-R3-0.75	15229
	5	0.25	densidad-media-R5-0.25	15611
		0.50	densidad-media-R5-0.50	15160
		0.75	densidad-media-R5-0.75	15160
	10	0.25	densidad-media-R10-0.25	15161
		0.50	densidad-media-R10-0.50	15229
		0.75	densidad-media-R10-0.75	15161
Alta		0.00	densidad-alta-NR	15276
	1	0.25	densidad-alta-R1-0.25	15279
		0.50	densidad-alta-R1-0.50	15277
		0.75	densidad-alta-R1-0.75	15276
	3	0.25	densidad-alta-R3-0.25	15277
		0.50	densidad-alta-R3-0.50	15277
		0.75	densidad-alta-R3-0.75	15275
	5	0.25	densidad-alta-R5-0.25	15276
		0.50	densidad-alta-R5-0.50	18282
		0.75	densidad-alta-R5-0.75	15278
	10	0.25	densidad-alta-R10-0.25	15277
		0.50	densidad-alta-R10-0.50	15276
		0.75	densidad-alta-R10-0.75	15276

ANEXO C

CÓDIGO DEL PROGRAMA GET DATA

```
args = commandArgs(trailingOnly=TRUE)

## Get command line arguments
folder <- args[1]
emm_file <- args[2]
map_file <- args[3]
t_ini <- as.numeric(args[4])
t_fin <- as.numeric(args[5])
step <- as.numeric(args[6])

wd <- paste("D:\\Tesis\\Procesamiento\\",folder, sep="")
setwd(wd)

cmd <- paste("awk -f ../getNodes.awk", emm_file, step, t_ini, t_fin, sep=" ")
system(cmd)

for(i in seq(t_ini,t_fin,step)){
  nodesFile <- paste("data/nodes",i,".txt",sep="")
  edgesFile <- paste("data/edges",i,".txt",sep="")

  cmd<-paste("../getEdges.exe -N", nodesFile, "-P", map_file, "-O", edgesFile, sep=" ")
  system(cmd)
}
```

ANEXO D

CÓDIGO DE LA FUNCIÓN GET UNIT INFO

```
get_unit_info <- function(folder, time, nodesFile, edgesFile) {
  ## Load data from txt files
  cat("Loading nodes and edges file for time ", time, "\n")
  nodes <- read.csv(nodesFile, sep = "", stringsAsFactors = FALSE)
  edges <- read.csv(edgesFile, sep = "", stringsAsFactors = FALSE)

  ## Create the base graph
  cat("Creating the base grap for time ", time, "\n")
  net <- graph_from_data_frame(d = edges[, c("orig", "dest")],
                               vertices = nodes$Id,
                               directed = F)

  V(net)$type = nodes$Type
  V(net)$label <- ""
  color_vertex <- terrain.colors(max(unique(V(net)$type)))
  V(net)$color <- color_vertex[V(net)$type]
  E(net)$weight <- -1 * edges[, "strengh"]
  l = as.matrix(nodes[, c("PosX", "PosY")])

  cat("Calculating metrics for time ", time, "\n")

  ## Number of nodes and edges
  V.nodes <- vcount(net)
  V.edges <- ecount(net)

  ## Degree centrality
  centr.deg <- centr_degree(net, mode = "all", normalized = T)
  deg <- centr.deg[[1]]
  deg.centr.value <- centr.deg[[2]]
  V(net)$size <- deg / max(deg) * 3 + 1
  V.deg.mean <- mean(deg)
  V.deg.max <- max(deg)
  V.deg.median <- median(deg)

  ## Edge density
  V.edge.density <- edge_density(net, 'F')

  ## Transitivity
  V.transitivity <- transitivity(net, type = "global")

  ## Diameter
  V.diameter <- diameter(net, directed = F, weights = NA)
  E(net)$color <- "gray60"
  E(net, path = V.diameter)$color <- "red"

  ## Mean distance
  V.mean.distance <- mean_distance(net, directed = F)

  ## Betweenness
  centr.betw <- centr_betw(net, directed = F, normalized = T)
  betw.centr.value <- centr.betw[[2]]
```

```
  ## Eigenvector
  E(net)$weight <- 120 - E(net)$weight
  centr.eigen <- centr_eigen(net, directed = F, normalized = T)
  eigen.centr.value <- centr.eigen[[4]]

  ## Plot the nodes based on diameter, type of bus (color), degree (size)
  cat("Creating graph image file for time ", time, "\n")
  png(
    file = paste("images/grafo-", folder, "-", time, ".png", sep = ""),
    height = 6910 / 2 ,
    width = 4250 / 2
  )
  plot(net, layout = l)
  dev.off()

  ## Community detection
  cfg <- cluster_fast_greedy(net)
  cfg.communities <- length(cfg)
  cfg.modularity <- modularity(cfg)
  cfg.crossing <- sum(crossing(cfg, net))

  ## Plot communities
    cat("Creating communities image file for time ", time, "\n")
    png(
      file = paste("images/comm-", folder, "-", time, ".png", sep = ""),
      height = 6910 / 2 ,
      width = 4250 / 2
    )
    plot(cfg, net, layout = l)
    dev.off()

  newRecord <- data.frame(
    scenario = folder,
    time = time,
    nnondes = V.nodes,
    nedges = V.edges,
    edge.density = V.edge.density,
    deg.mean = V.deg.mean,
    deg.max = V.deg.max,
    deg.median = V.deg.median,
    transitivity = V.transitivity,
    diameter = V.diameter,
    distance.mean = V.mean.distance,
    centr.deg = deg.centr.value,
    centr.betw = betw.centr.value,
    centr.eigen = eigen.centr.value,
    comm.number = cfg.communities,
    comm.modularity = cfg.modularity,
    comm.crossing = cfg.crossing
  )
  response = list(net, newRecord)
  return(response)
}
```

ANEXO E

CÓDIGO DEL PROGRAMA MAESTRO

```
@echo off
@setlocal

:: Save start time in a local variable
set start=%time%

:: Create path string concatenating the first command line arg with a base path
set @wd=D:\Tesis-Procesamiento\Procesamiento\%1

:: Delete existing folders in the working directory
echo Checking if folders exist
if exist %@wd%\data (echo Deleting data folder && rmdir /S /Q %@wd%\data)
if exist %@wd%\Rdata (echo Deleting Rdata folder && rmdir /S /Q %@wd%\Rdata)
if exist %@wd%\images (echo Deleting images folder && rmdir /S /Q %@wd%\images)

:: Create empty folders for saving nodes, edges, RData and created images from
echo Creating data folder && mkdir %@wd%\data
echo Creating Rdata folder && mkdir %@wd%\Rdata
echo Creating images folder && mkdir %@wd%\images
echo Folders are ready

:: Execute script to generate nodes and edges files
echo Executing getData.R
Rscript getData.R %*

:: Execute script to process data, generate graphs and calculate metrics
echo Executing processData.Rem
Rscript processData.R %*

:: Save end time in a local variable
set end=%time%

:: Aux variable to iterate times
set options="tokens=1-4 delims=:,"

:: Get hours, minutes, seconds and miliseconds from start and end times
for /f %options% %%a in ("%start%") do set start_h=%%a&set /a start_m=100%%b %% 100&set
/a start_s=100%%c %% 100&set /a start_ms=100%%d %% 100
for /f %options% %%a in ("%end%") do set end_h=%%a&set /a end_m=100%%b %% 100&set /a
end_s=100%%c %% 100&set /a end_ms=100%%d %% 100

:: Calculate difference between end and start time for hours, minutes, seconds and
miliseconds
set /a hours=%end_h%-%start_h%
set /a mins=%end_m%-%start_m%
set /a secs=%end_s%-%start_s%
set /a ms=%end_ms%-%start_ms%

:: Adjust values if difference was a negative value
if %ms% lss 0 set /a secs = %secs% - 1 & set /a ms = 100%ms%
if %secs% lss 0 set /a mins = %mins% - 1 & set /a secs = 60%secs%
if %mins% lss 0 set /a hours = %hours% - 1 & set /a mins = 60%mins%
```

```
if %hours% lss 0 set /a hours = 24%hours%
if 1%ms% lss 100 set ms=0%ms%

:: Show total execution time
set /a totalsecs = %hours%*3600 + %mins%*60 + %secs%
echo Excution time: %hours%:%mins%:%secs%.%ms% [%totalsecs%.%ms% sec(s) in total]
```

ANEXO F

ORGANIZACIÓN DE LOS ARCHIVOS

En el siguiente enlace se encuentran todos los archivos utilizados para la realización de este trabajo de titulación:

https://1drv.ms/u/s!AinvGjMuMPIRgahf-b4uYrc0BkU6HA?e=xynjhz

El primer directorio ***Conteo vehículos*** contiene la información del conteo de los vehículos y un archivo con el detalle de las horas seleccionadas para cada caso de densidad vehicular.

El siguiente directorio ***Simulaciones*** contiene una carpeta por cada escenario de simulación propuesto. Dentro de cada subdirectorio se tienen los archivos necesarios para ejecutar la simulación y obtener los reportes necesarios para la generación de los grafos.

El tercer directorio ***Procesamiento*** contiene los programas para el procesamiento de los datos tal como se detallaron en el Capítulo 2. Además, se encuentra un archivo *readme* con las instrucciones para obtener los archivos de nodos y aristas y a partir de estos obtener los grafos y sus métricas.

Se dispone de una carpeta de ejemplo para uno de los escenarios simulados, con la estructura y los archivos que se generan dentro de cada uno.

El último directorio ***Análisis resultados*** contiene las gráficas obtenidas para las diferentes métricas y las distintas formas de dibujar estas curvas.

ORDEN DE EMPASTADO