Universidad Central de Venezuela

Facultad de Ciencias

Computación

SIMULACIÓN MICROSCÓPICA DE TRÁFICO VEHÍCULAR BASADA EN AGENTES INTELIGENTES

Br. Daniel Alejandro Ampuero Anca

Br. Jesús Francisco Gómez Ortiz

Hector Navarro, Tutor

Caracas, 30 de mayo del año 2011

Universidad Central de Venezuela

Facultad de Ciencias

Computación

SIMULACIÓN MICROSCÓPICA DE TRÁFICO VEHÍCULAR BASADA EN AGENTES INTELIGENTES

Br. Daniel Alejandro Ampuero Anca

Br. Jesús Francisco Gómez Ortiz

Hector Navarro, Tutor

Caracas, 30 de mayo del año 2011

SIMULACIÓN MICROSCÓPICA DE TRÁFICO VEHÍCULAR BASADA EN AGENTES INTELIGENTES

Br. Daniel Alejandro Ampuero Anca
Br. Jesús Francisco Gómez Ortiz

Trabajo Especial de Grado presentado

ante la ilustre Universidad Central de Venezuela

como requisito parcial para optar al título de

Licenciado en Computación.

Hector Navarro, Tutor Fecha

Quienes suscriben, miembros del Jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado presentado por el **Br. Daniel Alejandro Ampuero Anca** y el **Br. Jesús Francisco Gómez Ortiz**, titulado: "**Simulación microscópica de tráfico vehícular basada en agentes inteligentes**" para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído como fue, dicho trabajo por cada uno de los miembros del Jurado, se fijó el día 30 de Mayo de 2011 a las 13:00 horas, para que sus autores lo defendieran en forma pública, lo que se hizo en la Sala PB I de la Escuela de Computacin en la Facultad de Ciencias, mediante una presentación oral de su contenido, luego de lo cual, respondieron las preguntas formuladas por el jurado y público en general. Finalizada la Defensa Pública del Trabajo Espacial de Grado, el jurado decidió aprobarlo por unanimidad.

En fe de lo cual se levanta la presente Acta, en Caracas a los treinta días del mes de mayo de dos mil once, dejándose también constancia de que actuó como Coordinador del Jurado el Profesor Tutor Héctor Navarro.

Hector Navarro, Tutor Fecha

Otilio Rojas Fecha

Oscar Anzola Fecha

Caracas, 30 de mayo del año 2011

A mi madre Asunción, sin ella nada hubiera sido posible

Daniel

A mi hijo Leonardo, mi fuente de inspiración, por su futuro

Jesús

Agradecimientos

Queremos agradecer a todas las personas que ayudaron al desarrollo de este trabajo especial de grado.

Agradecemos a Carlos Palumbo por abrirnos las puertas de la investigación al ponernos en contacto con ASICOR y URVISA.

Agradecemos a ASICOR y, especialmente, a Alfredo Galán por su cooperación y su prestancia a la hora de facilitarnos datos sobre la zona de estudio.

Agradecemos enormemente al Profesor Oscar Anzola por habernos ofrecido guía y sabios consejos en todo momento, su experiencia y sabiduría en temas de tráfico facilitó la resolución de muchísimos problemas.

Agradecemos a nuestro Tutor, el Profesor Héctor Navarro, por toda la ayuda y solidaridad prestada en momentos apremiantes, además del inmenso conocimiento aportado para la resolución de problemas relacionados con el visualizador de redes.

Agradecemos al Profesor Otilio Rojas por su visión aguda a la hora de encontrar errores de notación en los escritos de este documento.

Agradecemos al Profesor Eliezer Correa por brindarnos sus conocimientos en modelos matemáticos y simulación.

Agradecemos a Roberto Mateu por habernos ofrecido su visión y experiencia en el tema de la simulación de tráfico.

Agradecemos a Emilio Lazo por desarrollar y mantener la clase *ucv.cls* de LaTeXla cuál nos facilitó el trabajo de presentación del documento.

Finalmente, queremos agradecer a nuestras familias por estar a nuestro lado en cada momento y brindarnos su apoyo incondicional en los momentos más difíciles.

Resumen

La demanda de transporte, movida principalmente por las necesidades y deseos de movilización de las personas, y la oferta requerida para poder suplir esa demanda es un tema importante de estudio en ingeniería civil y planificación urbanística. El presente Trabajo Especial de Grado propone la utilización de un conjunto de herramientas para la construcción de estudios de tráfico sobre desarrollos urbanísticos actuales y a futuro apoyados en el *toolkit* MATSim, un simulador microscópico de tráfico basado en agentes inteligentes. Para probar la eficacia de este *toolkit* se realizaron un conjunto de pruebas con datos reales de la zona de Caracas de Los Cortijos de Lourdes y Los Ruices. Los resultados obtenidos develan que esta herramienta provee resultados utilizables y adaptables a las necesidades del analista de tráfico, generando resultados realistas con respecto a la dinámica vehicular de una zona particular.

Índice General

Introducción

El estudio de tráfico vehicular ha estado en desarrollo desde hace más de 75 años, pero no es sino en años recientes que ha tomado especial importancia, debido sobre todo al crecimiento exponencial de la población en las principales urbes del mundo y las consecuencias sobre el tráfico que este crecimiento genera. Una herramienta de gran poder para los planificadores urbanos hoy en día son los simuladores de tráfico que, con el incremento acelerado del poder de cómputo de las computadoras, se han convertido en una herramienta útil para el análisis y planificación urbana, además de constituir una extraordinaria fuente de investigación y desarrollo.

Diversos modelos matemáticos se han creado con la intención de capturar los aspectos fundamentales del tráfico. Estos modelos van desde la descripción del tráfico como variables relacionadas a la velocidad promedio, flujo y densidad vehicular, hasta modelos microscópicos de alto detalle implementados con agentes inteligentes.

El presente trabajo de tesis está dividido en diez capítulos, siendo los cinco primeros dedicados al planteamiento del problema y al marco teórico, en tanto que los últimos cinco se enfocan en los aspectos aplicativos del presente trabajo de tesis.

El capítulo 1 está dedicado al desarrollo del planteamiento del problema. En el capítulo 2 se realiza una descripción general del concepto de simulación, su tipología y usos. Además, se hace una revisión al proceso de simulación, haciendo énfasis en la metodología de desarrollo de un modelo de simulación, desde la formulación del problema hasta la recolección y análisis de datos devueltos por el simulador. El capítulo 3 introduce el concepto de agente inteligente y sus aplicaciones para la resolución de

tareas. Seguidamente, da cabida al concepto de sistemas multiagentes, presentando las peculiaridades y retos de estos sistemas, especialmente en lo que se refiere a interacción entre ellos. El capítulo 4 aborda los modelos de tráfico, realizando una categorización de estos y un repaso sobre las principales características de tráfico que son objeto de estudio en el modelo clásico de tráfico, para luego abordar algunos modelos microscópicos de tráfico, como el elemental modelo de seguimiento de carros. Por otro lado, se realiza también una revisión teórica del análisis de demanda de transporte, enfocándose en los cuatro pasos tradicionales de generación de demanda. Por último relaciona los conceptos de los modelos microscópicos de tráfico con los sistemas multiagentes. El capítulo 5 está dedicado al framework de simulación microscópica de tráfico MATSim, detallando su funcionamiento y uso.

El capítulo 6 refiere las técnicas y herramientas que se usaron durante el diseño y desarrollo de la aplicación. Los capítulos 7 y 8 abordan los detalles de diseño e implementación del visualizador de redes y el generador de demanda, respectivamente, así como también del proceso de recolección y generación de los datos de demanda por parte de URVISA. Finalmente, en los capítulos 9 y 10 se detallan los resultados obtenidos a partir de las simulaciones realizadas y las conclusiones obtenidas a partir de tales simulaciones.

Capítulo 1

Planteamiento del problema

1.1. Planteamiento del problema

En años recientes, los habitantes de las principales urbes del mundo han experimentado el crecimiento gradual y vertiginoso del congestionamiento vehicular en calles, avenidas, carreteras y autopistas. El congestionamiento ha pasado de ser un problema ocasional de ciertos instantes del día ("horas pico"), para convertirse en un molestia en cualquier momento del día, generando stress y contaminación, además de una considerable pérdida de tiempo productivo.

Venezuela no escapa de esta situación y muchas de sus grandes ciudades padecen de este problema. En Caracas, la mayor urbe del país, el asunto del tráfico se considera como uno de los principales problemas, puesto que diariamente transitan 1.550.000 vehículos en promedio, lo que representa 3 veces más carga vehicular que la que fue pensada para la ciudad [54]. Se estima que los desplazamientos en la ciudad toman cerca de una hora en promedio y más de 2 horas en las periferias [55].

El problema del tráfico genera múltiples problemas al ciudadano común y al estado venezolano, entre los cuales se encuentran:

- Pérdidas por concepto de depreciación y costes de mantenimiento de vehículos, las cuales, para el año 2007, rondaban en Bs. 3,8 millardos anualmente [57].

- Pérdida de productividad como consecuencia de los retrasos ocasionados por la congestión vehicular.
- Pérdidas por concepto de combustible, las cuales recaen principalmente en el estado venezolano puesto que el subsidio a la gasolina le cuesta anualmente alrededor de 4,2 millardos de dólares [59].
- Degradación de la calidad del aire de las grandes urbes debido a la emisión de gases contaminantes. Venezuela se encuentra en el puesto 28 en emisión de gases invernadero a nivel mundial [62].

1.2. Formulación del problema

Avizorando un escenario de deterioro del tráfico en Venezuela, y particularmente en las grandes urbes, se hace necesaria la búsqueda de una solución que permita aplicar medidas efectivas para el mejoramiento del tránsito vehicular. Entre algunas estrategias (algunas ya utilizadas) con posibles aplicaciones, encontramos:

- Mejoramiento y ampliación del sistema de subterráneo o Metro.
- Ampliación y mejoramiento de las vías urbanas e interurbanas.
- Reducción del tránsito vehicular mediante "Día de Parada".
- Integración del sistema de transporte superficial (autobuses) con el sistema Metro.
- Creación de canales especiales para el transporte público superficial.
- Disminución del subsidio a la gasolina y aumento del subsidio al transporte público.
- Estímulo al uso de medios de transporte alternativos que no incluyan automóviles.

Sin embargo, desde el punto de vista de la Ingeniería de Tránsito, es difícil prever y hacer una prospectiva de los resultados y consecuencias que cada una de las estrategias anteriores, aplicadas individualmente o en conjunto, causarán sobre el tráfico vehicular. Existen muchas herramientas de simulación que permiten a Ingenieros y Analistas preveer los efectos de la aplicación de algunas de las estrategias antes nombradas, sin embargo, con el incremento del poder de cómputo de los últimos años llama la atención los posibles estudios que se puedan realizar con herramientas de simulación microscópica.

Esto nos conduce a las siguientes interrogantes:

- ¿De qué forma una herramienta simulación microscópica permitirá a un Ingeniero o Analista de tránsito estudiar la dinámica vehicular de una zona y preveer los efectos de un posible cambio de la infraestructura vial?
- ¿Qué datos estadísticos relevantes se pueden obtener como resultado de realizar una simulación de tráfico?

1.3. Objetivo general

Simular la dinámica vehicular de la zona de Los Cortijos de Lourdes y Los Ruices a partir de datos reales de distribución de demanda de transporte, usando el software de simulación microscópica de tráfico vehicular basada en agentes inteligentes MATSim, con el fin de probar la validez y capacidad del software para la realización de estudios de tráfico en Caracas con la metodología de recolección de datos existente.

1.4. Objetivos específicos

- Realizar una compilación teórica sobre los modelos de simulación microscópica de tráfico, así como también de los modelos de generación, distribución y asignación de tráfico que permita una mejor comprensión del problema planteado.

- Diseñar e implementar una herramienta de visualización que facilite la creación y modificación de redes de tráfico en formato MATSim, así como la asignación de puntos de estaciones de conteo en la red.

- Diseñar e implementar una herramienta de visualización que permita el estudio de los resultados obtenidos en simulaciones, específicamente, los valores de volúmenes de tráfico asignados a las vías.

- Diseñar e implementar una herramienta de software que realice la tradución de datos reales de distribución de demanda obtenidos por estudios previos en la zona de Los Cortijos de Lourdes y Los Ruices, en forma de matrices origen destino, en datos de entrada válidos y utilizables por MATSim.

- Realizar simulaciones en MATSim a partir de los datos de distribución de demanda procesados a partir de matrices origen destino.

- Verificar y validar la salida generada por MATSim en las simulaciones, realizando los respectivos ajustes y calibraciones de la red y de los parámetros de simulación.

- Evaluar computacionalmente el desempeño de las simulaciones.

- Analizar los datos estadísticos obtenidos en las simulaciones realizadas y contrastarlos con datos recogidos por estudios previos.

1.5. Metodología de desarrollo

Para el desarrollo del modelo se usará la metodología de "Desarrollo con Prototipos" [81] [1] , puesto que permitirá observar el avance del proyecto en la forma de

[1]En Ingeniería de software el desarrollo con prototipación, pertenece a los modelos de desarrollo evolutivo, se inicia con la definición de los objetivos globales para el software, luego se identifican los requisitos conocidos y las áreas del esquema en donde es necesaria más definición. Entonces se plantea con rapidez una iteración de construcción de prototipos y se presenta el modelado (en forma de un diseño rápido). El diseño rápido se centra en una representación de aquellos aspectos del software que serán visibles para el cliente o el usuario final (por ejemplo, la configuración de la interfaz con el

pequeñas entregas, las cuales evolucionarán paulatinamente hacia el objetivo final del sistema. Por otro lado, permitirá el desarrollo de aquellas piezas de software de las que se tiene el conocimiento y las bases teóricas, relegando las piezas de las que poco se conoce a implementaciones posteriores, sin perder el sentido de la arquitectura del sistema. Además se hará uso el paradigma de "Programación Orientada a Objeto" (POO) como piedra angular del diseño e implementación.

usuario y el formato de los despliegues de salida). El diseño rápido conduce a la construcción de un prototipo, el cual es evaluado por el cliente o el usuario para una retroalimentación; gracias a ésta se refinan los requisitos del software que se desarrollará. La iteración ocurre cuando el prototipo se ajusta para satisfacer las necesidades del cliente.

Capítulo 2

Simulación

2.1. Definición de simulación

Muchos modelos pueden ser representados y resueltos de forma analítica. Una simple ecuación como:

$$\text{Distancia} = \text{Velocidad} \times \text{Tiempo}$$

es una solución analítica para el problema de un objeto que se mueve a una velocidad constante durante un período de tiempo. Para este tipo de modelos tan simples, la simulación no es necesaria. Sin embargo, muchísimos problemas del mundo real son usualmente más complejos. De hecho, son tan complejos que un modelo analítico simple no puede ser construido para representarlos. En este caso, el comportamiento del sistema debe ser estimado con una simulación. Algunas definiciones de simulación que se pueden encontrar en la literatura son:

> "Es una técnica numérica para llevar a cabo experimentos en una computadora digital, la cual involucra ciertos tipos de modelos matemáticos y lógicos que describen el comportamiento de un sistema (o subsistema) económico, de negocios, mecánico o físico a través del tiempo" [1]

> "Es la imitación de la operación de un sistema real a través del tiempo" [2]

> "Es el proceso de diseñar un modelo de un sistema real y llevar a cabo experimentos con este modelo, con el propósito de entender el comportamiento del sistema y/o evaluar varias estrategias para su operación" [3]

> "Una representación exacta de la situación a modelar es raras veces posible, limitando esta a aproximaciones con cierto grado de fidelidad, la cual es aceptable para propósitos de estudio del sistema." [4]

Se han construido modelos para casi cualquier sistema imaginable, incluyendo fábricas, telecomunicaciones, circuitos integrados, sistemas de autopistas y carreteras, dinámica de fluidos, economías nacionales, interacciones sociales, entre otras. En cada uno de esos ambientes, la simulación ha probado ser mucho más barata, segura y rápida que experimentar con el mundo real.

2.2. El proceso de simulación

El proceso de simular involucra muchos aspectos a tomar en cuenta. Dichos aspectos proporcionan, a la persona o grupo de personas que desean simular, una guía acerca de los pasos que se deben seguir para realizar una simulación correcta. Preguntas como:

- ¿Qué datos de entrada usar?
- ¿Qué información descartar y cuál tomar en cuenta en el modelo?
- ¿Cómo se valida el modelo construido?

Son interrogantes frecuentes a la hora de realizar una simulación. A continuación se describen los pasos más elementales que se deben realizar al momento de realizar una simulación [4]:

Definir el sub-espacio del problema: Lo primero que se debe hacer cuando se simula, es definir el problema que se va a tratar y los elementos que lo conforman. Eso incluye también la definición de los límites entre lo que incluirá la simulación y su ambiente, al igual que es importante definir el grado de precisión de los resultados que se esperan obtener. No se puede construir un modelo basado en definiciones vagas.

Definir el modelo conceptual: Una vez identificado el problema y sus límites, uno o varios modelos conceptuales apropiados pueden ser definidos. Estos incluyen los algoritmos que serán usados para describir el sistema, la entrada de datos requerida y los datos de salida a generar. Las suposiciones hechas sobre el sistema deben ser documentadas en esta fase, al igual que los efectos potenciales que estas suposiciones implican sobre los resultados o el grado de precisión del modelo. De igual forma, el modelo conceptual incluye los requerimientos de tiempo, personal y equipo que serán necesarios para elaborar y ejecutar el modelo. Todos los modelos potenciales deben ser evaluados y comparados en base a sus ventajas e inconvenientes, escogiendo al final el modelo que mejor se ajuste a las necesidades del proyecto.

Recolectar los datos de entrada: Una vez que el espacio del problema ha sido definido, se deben recolectar los datos necesarios para operar el modelo. Esto incluye información que pueda ser usada como parámetro de entrada, guía en el desarrollo de algoritmos y para evaluar el rendimiento de la simulación. Los datos de entrada deben contener información sobre el comportamiento conocido del sistema, al igual que información sobre las distribuciones estadísticas y variables aleatorias a ser usadas.

Construir el programa: Se construye el modelo de simulación en base a lo identificado en el Modelo Conceptual y a los datos recolectados.

Verificar, Validar y Acreditar el modelo (VV&A): Es esencial en el proceso de

simulación asegurarse que los algoritmos del modelo, los datos de entrada y las suposiciones hechas sobre el diseño son correctas y efectivamente resolverán el problema planteado. En esta fase del proceso se intentan identificar los posibles errores cometidos en las fases previas, antes de que el modelo se ponga en operación. Para los propósitos de la etapa VV&A, el proceso de desarrollo de una simulación es dividida en "Espacio del Problema", "Modelo Conceptual" y "Modelo de Software", cada una con transiciones definidas y así como también, evaluaciones de calidad en cada una de ellas, tal como se puede apreciar en la Figura 2.2

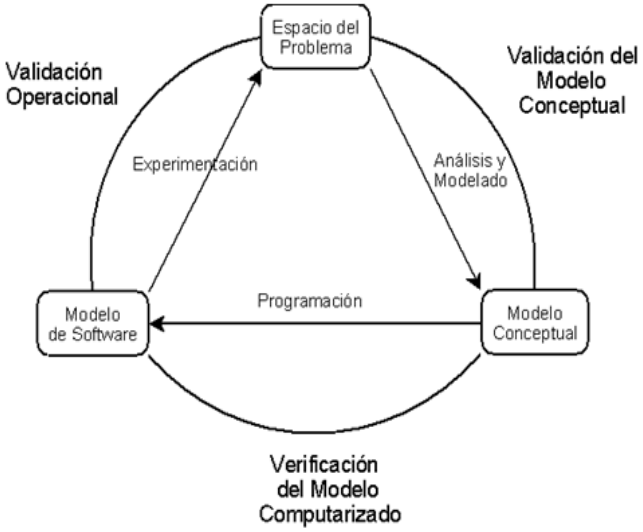


Figura 2.1: Etapa VV&A. Fuente [4]

De esta forma, se definen los conceptos Validación, Verificación y Acreditación:

Validación: Es el proceso de determinar si el modelo conceptual refleja los aspectos que necesitan ser atendidos en el espacio del problema y si tales requerimientos pueden ser alcanzados. También se ocupa de determinar si las operaciones del modelo de software final son consistentes con el mundo real, lo cual se realiza usualmente mediante experimentación y comparación con datos conocidos.

Verificación: Se ocupa de determinar si el modelo de software refleja de manera acertada el modelo conceptual sin errores de diseño o ejecución.

Acreditación: Es la aprobación del uso de un modelo de software para un propósito especifico. Un modelo de software acreditado para un propósito pudiera no serlo para otro.

Diseñar experimentos: Esta fase identifica los métodos más acertados y productivos para ejecutar la simulación y obtener las respuestas deseadas. Las técnicas estadísticas pueden ser usadas para diseñar experimentos que produzcan datos más acertados y con menos sesgo, con la menor cantidad de simulaciones posibles.

Ejecución de la simulación y obtención de datos: En esta fase se pone en marcha el modelo de software construido, recogiendo tras cada simulación los datos resultantes, los cuales serán organizados, categorizados y almacenados. Es posible que se realicen cambios en los datos recogidos sin necesidad de cambiar el modelo conceptual o el algoritmo. En el caso de los modelos de Monte Carlo, quizás cientos o miles de ejecuciones serán necesarias para alcanzar resultados confiables.

Analizar los datos obtenidos: Los datos obtenidos durante las simulaciones pueden ser bastante voluminosos y esparcidos a través del tiempo, por lo que es necesario realizar un análisis detallado, de manera de poder obtener respuestas a las interrogantes que motivaron la simulación. El análisis puede producir información en forma de tablas, gráficos, mapas, animaciones, entre otros.

Documentar los resultados: Los resultados de los estudios realizados con la simulación deben ser documentados para su subsecuente divulgación.

Capítulo 3

Agentes inteligentes

3.1. Definición de agente

> "Un agente es todo aquello que puede considerarse que percibe su ambiente mediante sensores y que responde o actúa en tal ambiente por medio de efectores." [15]

> "Un agente es un sistema computacional situado en determinado ambiente, el cual es capaz de realizar acciones autónomas en este ambiente con el fin de alcanzar los objetivos para los cuales fue diseñado." [16]

> "Los agentes autónomos son sistemas computacionales que habitan algún ambiente dinámico, tienen percepción y actúan de forma autónoma en este ambiente, satisfaciendo de esta forma un conjunto de metas o tareas para las cuales fue diseñado." [17]

Existen varias definiciones de agentes en la literatura, pero todas concuerdan al menos en algunas de las siguientes características:

- Un agente tiene comportamiento autónomo
- Un agente tiene sensores que permiten captar su ambiente
- Un agente actúa y modifica el ambiente donde habita con el fin de alcanzar sus metas o cumplir las tareas para las cuales fue diseñado

Una definición que conjuga bien estos tres planteamientos es:

> "Un agente autónomo es un sistema que está situado dentro o es parte de un ambiente, el cual percibe y actúa sobre él a través del tiempo para la consecución de su propias metas (agenda), afectando futuras percepciones sobre este ambiente." [18]

Por otro lado, se encuentra la siguiente definición de agente racional de la mano de Russell y Norving:

> "Por cada secuencia de percepciones, un agente racional debe hacer lo que sea que maximice su medida de rendimiento, con base en la evidencia provista por la secuencia de percepciones y con cual sea el conocimiento que fue incorporado en el agente" [15]

La Figura 3.1 presenta el diagrama básico de un agente interactuando con su ambiente a través de sus sensores y efectores. Es posible observar también la naturaleza cíclica de la interacción, puesto que el ambiente sobre el que el agente actúa y afecta, es captado de nuevo a través de sus sensores. Es decir, el agente es capaz de modificar su ambiente y percibir tal cambio.

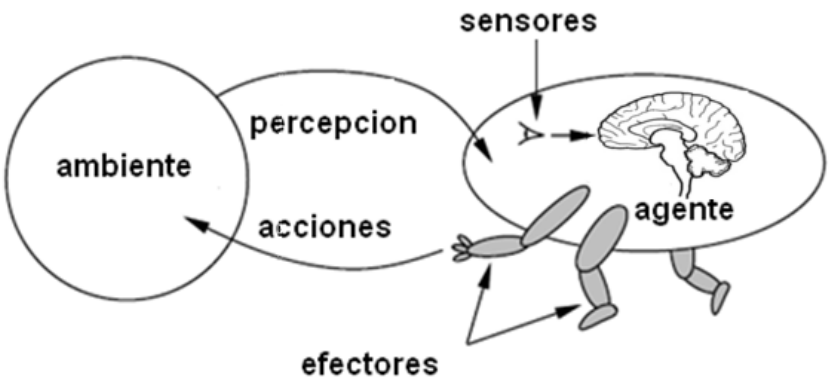


Figura 3.1: Agente y su interacción con el ambiente Fuente: [15], p. 32

Así se tiene que un sistema basado en agentes es un sistema donde la abstracción clave usada son los agentes y sus interacciones con el medio ambiente [16].

3.2. Tipos de agentes

Se consideran cuatro diferentes tipos de agentes [15]:

Agentes de reflejo simple: Este tipo de agente funciona con un mecanismo denominado regla condición-acción, también denominado producciones y regla situación-acción, el cual pudiera describirse en el caso de un agente simple que responda ante un saludo con "Hola, ¿cómo estás?" de la siguiente forma:

```
si saludo entonces responder "¿Hola cómo estás?"
```

Este tipo de agentes es el más simple de todos, puesto que sólo dependen de percepciones inmediatas.

Agentes bien informados en todo lo que pasa: Este tipo de agentes, a diferencia de los agentes de reflejo simple, mantienen un estado interno con información que obtuvieron de sus sensores en oportunidades pasadas, otorgándole al agente una memoria de cómo el ambiente se ha ido alterando a través del tiempo y de sus acciones.

Agentes basados en metas: Este tipo de agentes no sólo basan sus acciones en reflejos simples o en modelos del mundo, sino que también están impulsados por el cumplimiento de metas propias. Un agente basado en metas tomará una decisión particular por encima de otra, aún cuando ambas sean igualmente factibles, si esta satisface mejor sus metas.

Agentes basados en utilidad: Supongamos que varias secuencias de decisiones llevan a la consecución de las metas de un agente. En este caso, nos enfrentamos a un dilema acerca de cuál secuencia de decisiones es mejor para los fines del agente. Es en este punto donde entran en acción los agentes basados en utilidad,

los cuales, desde su punto de vista como agentes, hacen la siguiente consideración: "¿Qué tanto me satisface este estado?". De esta forma, se puede establecer mediante una función de utilidad una ponderación de satisfacción con cada estado. La especificación de esta función de utilidad, permite la resolución de dos problemas:

Conflicto de metas: es posible que algunas metas sean incompatibles o generen conflictos, por lo que sólo algunas de ellas se pueden obtener. Una función de utilidad ayudará a definir cuáles metas se deben lograr a fin de optimizar la satisfacción.

Discriminación de metas: Si existen varias metas que el agente desea cumplir, pero ninguna es alcanzable con certeza, entonces la utilidad provee una forma de ponderar positivamente las metas alcanzables en detrimento de aquellas que son más importantes pero inalcanzables.

3.3. Tipos de ambientes

Las principales clasificaciones de ambientes donde coexisten los sistemas basados en agentes son los siguientes [15]:

Accesible: Si el agente es capaz de percibir completamente el estado del ambiente, se dice entonces que este ambiente es accesible. Un ambiente es accesible efectivamente si los sensores del agente perciben todos los aspectos relevantes para la toma de decisiones de éste. Los ambientes accesibles son convenientes y deseables pues de este modo el agente no debe mantener ningún estado interno para hacerle seguimiento al ambiente.

Determinista: Si el siguiente estado del ambiente es determinado completamente por el estado actual y las acciones escogidas por el agente, entonces estamos en presencia de un ambiente determinista. En principio un agente no debe preocuparse

acerca de la incertidumbre en un ambiente accesible y determinista. No obstante, si el ambiente es inaccesible, éste pudiera parecer no determinista. Esto es particularmente cierto cuando el ambiente es complejo, haciendo difícil mantener registro de todos los aspectos inaccesibles. Por ende es preferible estipular si un ambiente es determinista o no determinista desde el punto de vista del agente.

Episódico: En un ambiente episódico, la existencia de un agente pudiera ser dividida en episodios. Cada episodio consiste en la rutina "percepción-acción", donde el agente percibe su ambiente y luego actúa en consecuencia. Las acciones tomada durante un episodio son independientes de las que fueron tomadas en episodios pasados. Los ambientes episódicos son más simples que los no episódicos, puesto que el agente no necesita deliberar por posibles eventos futuros.

Estático: Si el ambiente puede cambiar mientras el agente delibera, se dice entonces que el ambiente es dinámico para ese agente, de otra forma se considera que estático. Los ambientes estáticos son mucho más simples que los dinámicos, puesto que el agente no necesita percibir su ambiente mientras decide sus acciones, ni tampoco necesita tomar en consideración el paso del tiempo.

Discreto: Si existen un número de distintas y claramente definidas percepciones y acciones, entonces se dice que el ambiente es discreto.

3.4. Sistemas multiagentes

> "En general, los sistemas multiagentes son sistemas computacionales en los cuales varios agentes semi-autónomos interactúan o trabajan juntos para realizar algún conjunto de tareas o satisfacer algún conjunto de metas. Estos sistemas pueden involucrar agentes computacionales que sean homogéneos o heterogéneos y cuyas metas pueden ser comunes o no comunes." [19]

> "Un sistema (multiagente) contiene un número de agentes que interactúan unos con otros comunicándose. Los agentes son capaces de actuar en un ambiente; diferentes agentes tienen diferentes "esferas de influencia", en el

> sentido que ellos tendrán control – o al menos serán capaces de influenciar – sobre diferentes partes del ambiente." [16]

Se pueden encontrar muchos ejemplos de sistemas multiagentes en la naturaleza. Un ejemplo sencillo es una colonia de hormigas, donde el comportamiento de cada individuo es bastante simple y son las interacciones grupales las que le dan forma y dinamizan el sistema. Los sistemas multiagentes, más allá de resolver problemas cotidianos, nos permite observar la dinámica de sistemas complejos integrados por varios agentes con comportamientos que pueden ser muy simples o altamente complejos.

3.4.1. Interacción en sistemas multiagentes

Un enfoque común para modelar agentes inteligentes es usar agentes basados en utilidad (véase 3.2 Tipos de agentes). Este tipo de enfoque permite que los agentes tengan preferencias con una decisión (en base a los resultados obtenidos) sobre otras [20].

Sin embargo, cuando hay varios agentes involucrados, la toma de decisiones en base a las preferencias se torna algo más complicado. Para simplificar las cosas, se asumirá que se tienen dos agentes denominados A y B. Se asume además que cada agente tiene sus propios intereses y vela por ellos. Ambos agentes tienen un conjunto O de salidas ó estados sobre los cuales los dos agentes tienen preferencia, que concretamente pueden verse como resultados de un juego en que estén participando. Estas preferencias se capturan usando una función de utilidad u la cual asigna un número real a cada resultado [21]:

$$u : O \to R$$

Puesto que cada resultado se relaciona con un escalar, es posible crear una ordenación de preferencia sobre los resultados. Sea O el conjunto de los posibles resultados de tomar una decisión (salidas), se definen las siguientes relaciones:

a Para cualquier par o_1 y $o_2 \in O$, se denota $o_1 > o_2$ como que el agente prefiere estrictamente o_1 sobre o_2.

b Para cualquier par o_1 y $o_2 \in O$, se denota $o_1 \geq o_2$ como que el agente prefiere débilmente o_1 sobre o_2.

c Para cualquier par o_1 y $o_2 \in O$, se denota $o_1 \sim o_2$ como que el agente es indiferente entre o_1 y o_2.

La relación de preferencia débil tiene además las siguientes propiedades [21]:

Reflexividad: para todo $o \in O,\ o \geq o$

Transitividad: si $o_1 \geq o_2$ y $o_2 \geq o_3$ entonces $o_1 \geq o_3$

Comparable: pata todo $o_1 \in O$ y $o_2 \in O,$ se pude cumplir $o_1 \geq o_2$ ó $o_2 \geq o_1$

La relación de preferencia estricta cumple con la segunda y tercera propiedad, sin embargo es claro que no cumple con la primera. Es posible plantear una relación entre la incertidumbre y las preferencias de un agente, para lo cual se usa el concepto de *lotería*. Una lotería es una selección aleatoria de una de las decisiones en base a probabilidades específicas. Las probabilidades de tomar una u otra decisión están en base a las preferencias, donde las decisiones con más preferencia tienen mayor probabilidad de ser escogidas [20].

En un modelo multiagente, las decisiones tomadas por un agente afectan los resultados de las decisiones tomadas en simultáneo por el otro agente. Para simplificar esto, se asumirá que existen dos posibles estrategias[1] para los agentes A y B: Cooperar o No Cooperar (C, N). Se denominará a este conjunto de estrategias como E. Esta simplificación permite reducir el conjunto de salidas a una función:

[1]Una estrategia es un plan completo de acción para cualquier situación que surja, la cual determina completamente el comportamiento del agente. No se debe confundir el término estrategia con "movimiento", el cual es una acción tomada por el agente en un momento determinado.

$T : E_A \times E_B \rightarrow O$

Puesto que el conjunto O producido por la función T es finito, es posible saber de antemano qué utilidad obtendrá cada agente en cada situación. La interrogante que sigue es:

> ¿Si fueras el agente A dado un escenario cualquiera, qué escogerías: Cooperar o No Cooperar?

3.4.2. Juegos, estrategias y Equilibrio de Nash

Se define un juego formal como [20]:

> Una tupla (N, A, u), donde:
> - N es el conjunto finito de n jugadores, indexados por i;
> - $A = A_1 \times \ldots \times A_n$, donde A_i es el conjunto finito de acciones disponibles para el jugador i. Cada vector $a = (a_1, \ldots, a_n) \in A$ es llamado perfil de acción;
> - $u = (u_1, \ldots, u_n)$, asociado a cada perfil de acción, donde $u_i : A \rightarrow R$ es una función de utilidad para el jugador i.

Se hace la suposición implícita que $O = A$ puesto que cada perfil de acción tiene asociado una única salida.

De ahora en adelante, en conformidad con la nomenclatura usada en Teoría de Juegos, las *acciones* a serán llamadas *estrategias* s, y el conjunto de todas las posibles estrategias será denominado S.

Se definen dos tipos de estrategias [20]:

Estrategia pura: Dado el conjunto de estrategias posibles, el jugador toma una de ellas y juega acorde a esta.

Estrategia mixta: Dado el conjunto de estrategias posibles, el jugador toma aleatoriamente alguna de ellas de acuerdo a una distribución de probabilidad específica.

Sin importar qué tipo de estrategia se escoja, un *perfil* de estrategia es definido como un conjunto de estrategias, donde todos los jugadores están involucrados y existe una sola estrategia para cada uno. Se define el conjunto de los *perfiles de estrategia* como el producto cartesiano de las estrategias de cada jugador [20]. Es claro ver que cada *perfil de estrategia* supone una salida, y por consiguiente, una utilidad para cada agente.

De vuelta a la interrogante planteada en la sección anterior, se introducirá el concepto de dominancia de estrategia, el cual será clave en la resolución de ésta. Se define la dominación (de estrategia) Paretto como [20]:

> "Un perfil de estrategia s tiene una dominación Paretto sobre un perfil de estrategia s', si para cada $i \in N$ se cumple que $u_i(s) \geq u_i(s')$ y existe algún $j \in N$ tal que $u_j(s) \geq uj(s')$"

La dominación Paretto permite definir una relación de orden parcial sobre los perfiles de estrategia. Sin embargo, no es posible distinguir la estrategia óptima, sino un conjunto de ellas. De esta forma, se define la Optimalidad Paretto como [20]:

> "Un perfil de estrategia s es Óptimo Paretto o Estrictamente Eficiente Paretto, si no existe ningún perfil de estrategia que tenga dominación Paretto sobre s."

Se define $s_{-i} = (s_1, s_2, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ formalmente como el perfil de estrategia s sin contemplar la estrategia del jugador i. Por lo tanto, cualquier estrategia s puede ser escrita como $s = (s_i, s_{-i})$. Si los jugadores distintos a i jugaran s_{-i}, un jugador que espere maximizar su utilidad enfrentará el problema de determinar su mejor respuesta, la cual es definida como [20]:

> "La mejor respuesta a un perfil de estrategia s_{-i}, es una estrategia mixta $s_i^* \in S_i$ tal que $u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i})$, para todas las estrategias $s_i \in S_i$."

La mejor respuesta no siempre es única, el número de mejores respuestas es infinito, excepto en el caso que la mejor respuesta sea una estrategia pura. En general, un jugador no sabrá qué estrategias piensan adoptar el resto de los jugadores, aún cuando puede tener conocimiento de sus posibles estrategias. Por ende, el concepto de *mejor respuesta* no ofrece soluciones, pues no identifica qué estrategia debe tomar un jugador. Sin embargo, este concepto permitirá dar paso a un concepto de gran importancia en teoría de juegos no cooperativa y es el de "Equilibrio de Nash" [20]:

> "Un perfil de estrategia $s = (s_1, \ldots, s_n)$, es un Equilibrio de Nash si, para cada agente i, s_i es la mejor respuesta a s_{-i}"

El Equilibrio de Nash implica entonces que [21]:

- Ningún jugador quisiera cambiar su estrategia, si supiera qué estrategia siguen el resto de los jugadores.
- Suponiendo que todos los jugadores excepto i jugaran lo mejor posible, i jugará lo mejor posible, aún cuando signifique no obtener la máxima utilidad posible.

Capítulo 4

Modelado de Tráfico

El tráfico vehicular ha sido tema de estudio desde hace más de 70 años, siendo iniciado con las investigaciones de Greenshields sobre la dinámica real del tráfico [26]. Greenshiled desarrolló un modelo lineal del flujo del tráfico ininterrumpido, el cual se basa en la suposición que la velocidad y la densidad vehicular están linealmente relacionadas mediante la siguiente fórmula [27]:

$$q = k\left(u - \frac{u^2}{u_f}\right)$$

Donde:

$$\begin{aligned} q &= \text{Flujo vehicular} \\ k &= \text{Densidad vehicular} \\ u &= \text{Velocidad} \\ u_f &= \text{Velocidad de flujo libre} \end{aligned}$$

El modelo de Greenshields dominó el campo de investigación sobre Teoría de Flujo de Tráfico durante más de 50 años [27], aún cuando presenta varios problemas relacionados a cuán exacto modela las situaciones de tráfico. Desde entonces, se han desarrollado varios modelos de tráfico que pretenden modelar ciertas características particulares de la dinámica vehicular.

4.1. Clasificación de los modelos de tráfico

En función a su alcance, se distinguen tres categorías [27]:

Microscópico: Se modela el tráfico vehicular a partir de cada uno de los individuos que componen su dinámica, pudiendo ser estos automóviles, peatones o motos. Se hace especial énfasis en los detalles, bajo el supuesto que la interacción entre cada una de las partes compone la globalidad del tráfico.

Macroscópico: Se modela la dinámica vehicular a partir de medidas tales como el flujo vehicular, la densidad vehicular y/o la velocidad. A diferencia de los modelos microscópicos, este tipo de modelos no toma en cuenta a cada automóvil, sino que considera a todos o a un grupo de automóviles como un solo ente, con características distintas en cada trayecto de una carretera, autopista o calle.

Mesoscópico: A menudo llamados "modelos de transición", conjugan las características de los modelos microscópicos con las características de los modelos macroscópicos.

Es posible apreciar gráficamente las diferencias entre los modelos macroscópicos y microscópicos en la Figura 4.1.

Es posible dividir también los modelos de tráfico en función a la ubicación espacial de los vehículos en dos categorías [27]:

Discretos: Similar a los autómatas celulares, la carretera es divida en celdas de igual tamaño y cada vehículo ocupa sólo una a la vez en un momento dado. De igual forma, el transcurrir del tiempo se hace "a trozos".

Continuos: Cada vehículo, como sucede en la realidad, ocupa una posición (x, y, z). Igualmente, el flujo del tiempo es continuo en estos modelos.

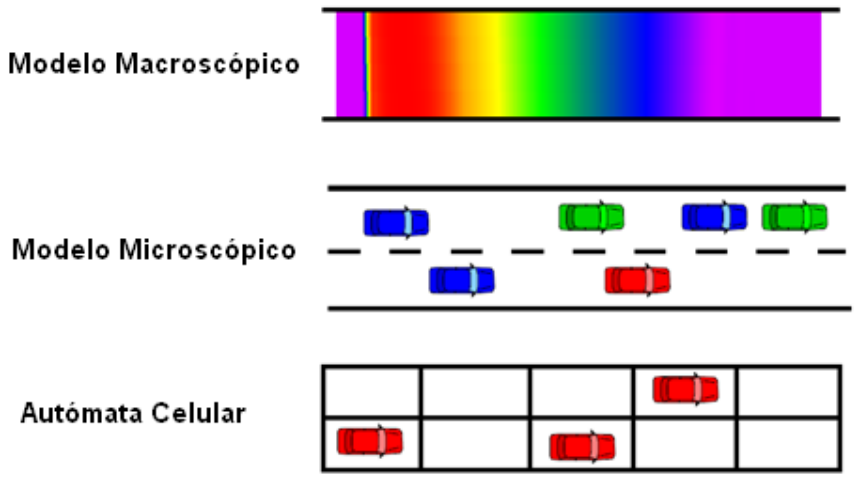


Figura 4.1: Comparación entre las distintas aproximaciones al modelado de tráfico. Fuente [29], p.330

También se pueden clasificar los modelos en base a la situación que modela. Las dos situaciones más comunes son [27]:

Intersecciones: Caracterizado por tener una dinámica vehicular menos extensa, pero más intensa, pues intervienen distintos factores a modelar tales como: cambios de carril, semáforos, múltiple aceleración y desaceleración en un período de tiempo corto, colisiones.

Autopista/carretera: De mayor escala, se suele modelar con modelos macroscópicos.

4.2. Diagrama básico del tráfico: variables de interés

En el modelo de tráfico iniciado por Greenshields y, continuado y extendido principalmente por Lighthill y Whitman (1955), se suele hacer especial énfasis en el estudio y análisis de tres características del tráfico. Estas son [27]:

Tasa de flujo: la tasa de flujo q, es el número N de vehículos contados que entran en un segmento de vía, dividida entre el tiempo T que duró la medición.

$$q = \frac{N}{T}$$

Esta medida es usualmente expresada en términos de vehículos por hora, aún cuando el tiempo T de medida generalmente ha sido mucho menor a una hora.

Velocidad: La velocidad instantánea de un vehículo individual se modela como:

$$u_i = \frac{\mathrm{d}x}{\mathrm{d}t} = \lim_{t_2 - t_1 \to 0} \frac{x_2 - x_1}{t_2 - t_1}$$

Sin embargo, para el cálculo de velocidad media a partir de una muestra[1], se siguen dos métodos. El primero es una simple media aritmética de las muestras observadas:

$$\bar{u_t} = \frac{1}{N} \sum_{i=1}^{N} u_i$$

Donde u_i es la velocidad de cada muestra y N la cantidad de muestras.

El segundo método es denominado *"velocidad media espacial"*, de la cual hay, por desgracia, varias definiciones que no son equivalentes. Una definición propuesta por Wardrop en 1952 [71] se basa en el tiempo promedio que le toma a un vehículo cruzar una distancia dada D:

$$\bar{u_s} = \frac{D}{\frac{1}{N} \sum_i^N t_i}$$

Donde t_i está definido como el tiempo que le toma al vehículo i recorrer la distancia D:

$$t_i = \frac{D}{u_i}$$

[1] El método de muestreo puede variar, desde medición por radar, hasta el antiguo método de medir el tiempo que le toma a un vehículo desplazarse a través de un pequeño segmento de vía

Lo que permite simplificar la ecuación:

$$\bar{u_s} = \frac{D}{\frac{1}{N}\sum_i t_i} = \frac{1}{\frac{1}{N}\sum_i \frac{1}{u_i}}$$

El segundo método consiste en tomar el promedio de velocidad de todos los vehículos en un segmento de vía en un instante dado. Se puede ver como una "fotografía aérea" donde se asume que cada vehículo lleva el velocímetro en el techo. Una forma más realista de ver esta situación consiste tomar dos fotos aéreas del mismo segmento con una ligera diferencia de tiempo, apreciando de esta forma la velocidad de cada vehículo.

Densidad: La densidad, que representa la cantidad de vehículos por unidad de longitud, puede ser calculada a partir de la velocidad o el flujo mediante la siguiente relación:

$$k = \frac{q}{\bar{u_s}}$$

Esta medida es usualmente expresada en términos de vehículos por kilómetro.

La relación que forman entre sí el flujo y la densidad puede ser observada en la Figura 4.2

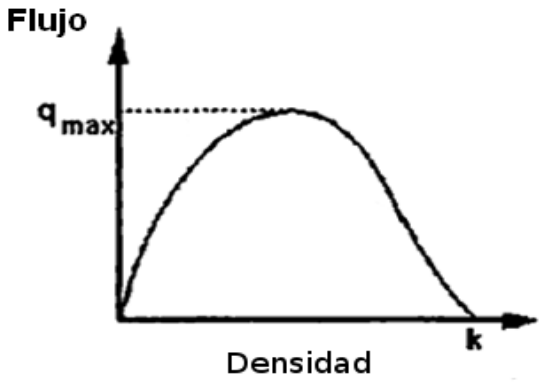


Figura 4.2: Diagrama Fundamental (Flujo/Densidad) Fuente: [28], p.5

De igual forma, la relación entre la velocidad y la densidad es ilustrada en la Figura 4.2

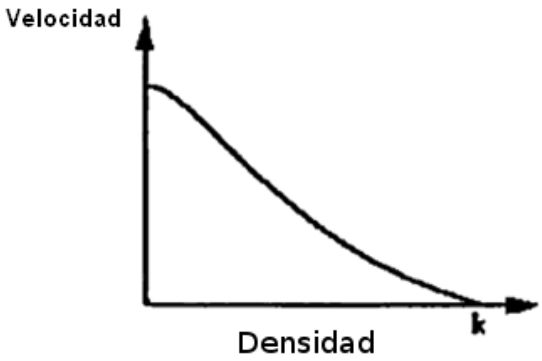


Figura 4.3: Diagrama Fundamental (Velocidad/Densidad) Fuente: [28], p.5

4.3. Modelos microscópicos de tráfico

La tarea relativamente sencilla y común de un vehículo siguiendo a otro en una vía, donde no hay cambio de canal o giros, puede ser categorizada en tres subtareas específicas [30]:

Percepción: El conductor recolecta la información de su entorno, tal como la distancia y velocidad de los vehículos vecinos, así como su propia velocidad.

Toma de Decisiones: En base a la información recolectada y el conocimiento previo, el conductor toma una o varias decisiones.

Control: Una vez tomada la decisión, ésta es puesta en marcha.

Adicionalmente, considerando la respuesta del conductor al ambiente, su comportamiento puede ser clasificado en tres categorías, cada una con mayor detalle [31]:

- Estratégica

- Táctica
- Operacional

La figura 4.4 ofrece una visión general de los principales factores y características que influyen en la conducción humana.

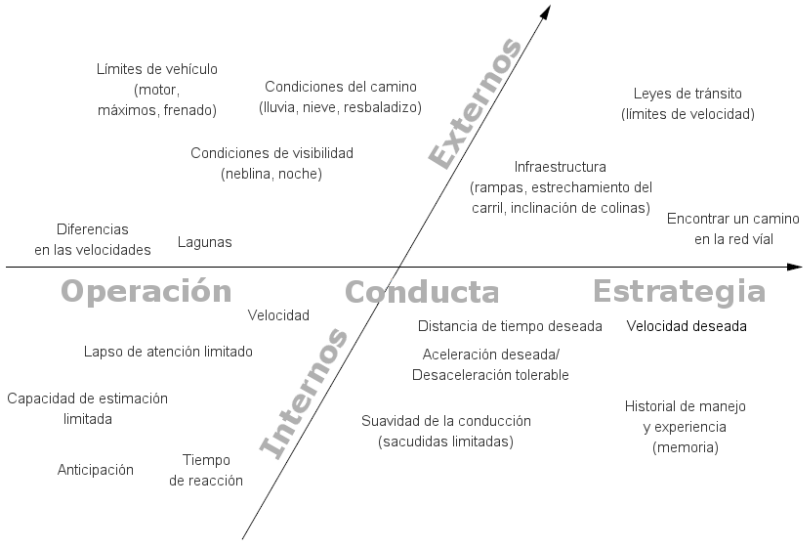


Figura 4.4: Principales factores y características que influyen en la conducción Fuente: [29], p.331

Sin embargo, no está claro cómo los conductores llevan a cabo estas funciones en detalle. Los intentos de relacionar matemáticamente los estímulos que recibe un conductor con las decisiones y las acciones tomadas en consecuencia han tenido un éxito modesto [30]. Una de las dificultades principales es que el conductor no tiene una única función de transferencia[2], puesto que se comporta de manera distinta bajo condiciones diferentes [30]. Sin embargo, los modelos microscópicos clásicos de tráfico no toman en cuenta todos estos factores, interesándose sólo en algunos de ellos [30].

[2]En el ámbito de los modelos microscópicos, una función de transferencia para un conductor es una función que toma como entrada un estímulo y retorna una acción.

4.3.1. Modelos de seguimiento de carros ("Car Following Models")

Los primeros modelos de seguimiento de carros y a la vez los primeros modelos microscópicos de tráficos, fueron propuestos independientemente por Reuschel en 1950 y Pipes en 1953 [26]. Ambos modelos están inspirados en las reglas de conducción basadas en las reglas de distancias entre vehículos contempladas en los reglamentos de tráfico, eso es, mantener cierta distancia con el vehículo siguiente proporcional a su velocidad [28]. En los modelos de Reuschel y Pipes se considera que la respuesta de los conductores ante los estímulos se realiza de forma inmediata [28]. Poco después, Herman y sus colaboradores en el Laboratorio de Investigación de General Motors para el año 1957 desarrollaron un modelo ligeramente distinto, en el cual se toma en cuenta el tiempo de reacción de los conductores e incluye como variable de estudio el control de la aceleración por el conductor [28].

La ecuación básica en el modelo de Seguimiento de Carros es la siguiente:

$$\frac{\mathrm{d}^2 x_{n+1}(t+T)}{\mathrm{d}t^2} = \lambda \left[\frac{\mathrm{d}x_n(t)}{\mathrm{d}t} - \frac{\mathrm{d}x_{n+1}(t)}{\mathrm{d}t} \right]$$

Donde:

$$\begin{aligned} n &= \text{Carro líder} \\ n+1 &= \text{Carro seguidor} \\ T &= \text{Retardo de tiempo de reacción} \\ \lambda &= \text{Coeficiente de reacción del conductor ante un estímulo} \\ &\quad \text{(coeficiente de sensibilidad)} \end{aligned}$$

En los primeros estudios de seguimiento de carros, el coeficiente de sensibilidad λ era asumido constante [28].

Herman y sus colegas probaron su modelo llevando a cabo experimentos, investigando las consecuencias de este modelo en la estabilidad del flujo vehicular [28]. La estabilidad se refiere a la habilidad de un sistema de vehículos para absorber una perturbación introducida en el tráfico. La estabilidad local se refiere a la posición relativa de un par de vehículos. La estabilidad *asintótica* se refiere a la atenuación o incremento de la amplitud de la perturbación mientras se mueve a través de una línea de vehículos que circulan en dirección a la fuente de la perturbación [28].

4.3.2. Modelos de colas

En simulaciones microscópicas de gran escala suele ser necesario modelos más simples debido a las limitaciones de tiempo y espacio. Una solución a tales limitaciones es el uso de modelos basados en colas [65].

En el modelo de colas propuesto por Gawron [66] cada enlace (segmento de vía) es un sistema de colas de prioridad con los siguientes atributos:

- Velocidad de flujo libre v_0
- Longitud L
- Capacidad C
- Número de Canales n_c

El tiempo de viaje a velocidad de flujo libre es calculada de la forma $T_0 = L/v_0$. La constante de almacenamiento de un enlace es calculada como $N_s = L{\cdot}n_c/l$, donde l es el espacio que ocupa un vehículo en metros.

En la lógica de intersecciones en el modelo de Gawron [66], los enlaces son procesados en un orden arbitrario pero fijo. Un vehículo es movido al siguiente enlace si [65]:

1. Ha llegado al final del enlace: un vehículo que entra en el enlace a en tiempo t_0 no puede dejar el enlace antes del tiempo $t_0 + T_0$, donde T_0 es el tiempo para recorrer del enlace en velocidad de flujo libre de flujo.

2. Puede ser movido de acuerdo a la capacidad del enlace siguiente: la condición es determinada por:

$$N < C \text{ ó } (N = C \text{ y } rnd < f)$$

   donde C es la parte entera de la capacidad del enlace (en vehículos por *step* de tiempo), f es la parte fraccional de la capacidad del enlace, y N es la cantidad de vehículos que abandonaron el enlace en el mismo *step* de tiempo. rnd es un número aleatorio tal que $0 \leq rnd \leq 1$. De acuerdo con esta fórmula, un vehículo puede abandonar el enlace si la capacidad de salida del enlace no ha sido agotada para este *step* de tiempo. Si la capacidad por *step* de tiempo no es un entero, entonces se mueve el último vehículo de acuerdo a una probabilidad que es igual a f.

3. Existe espacio en el enlace siguiente: si el enlace destino está lleno entonces el vehículo no se moverá.

4.4. Análisis de demanda de transporte

La demanda de transporte, de forma diferente a la demanda de otros productos, es una demanda derivada [67], i.e. una persona necesita transportarse para suplir una necesidad secundaria, tal como asistir al trabajo, la escuela, compras, cine, etc. Por lo tanto, dos aspectos de gran importancia en el análisis de demanda son la zonificación y los propósitos de los viajes.

La zonificación se refiere al patrón de uso de la tierra en un área. La generación y la distribución de los viajes se ven directamente afectados por la zonificación puesto que ciertos tipos de zonas producirán o atraerán vehículos en ciertas horas del día. Por

ejemplo, una zona residencial producirá volumen vehicular en las horas de la mañana, mientras que la zonas comerciales e industriales la atraerán en horas matutinas.

Por otro lado, los propósitos de los viajes determinan otros aspectos tales como la selección de la zona donde se viajará y el tiempo de permanencia en esa zona. Por ejemplo, si el propósito del viaje es ir a trabajar, entonces la duración de este viaje está determinado por la cantidad de horas laborales. Igualmente, el lugar de trabajo no es algo que deba escogerse cada día, sino que es fijo. En cambio, si el viaje es recreacional una persona escoge más libremente a dónde ir y cuánto tiempo permanecer realizando la actividad.

Resumiendo, el proceso de decisión para realizar un viaje está compuesto por los siguientes pasos [67]:

- Decisión de viajar.
- Escogencia del destino.
- Escogencia del modo de transporte.
- Determinación de la ruta de viaje.

Cada uno de los cuatro pasos anteriores está relacionado directamente con los pasos del proceso tradicional de generación de demanda [68]:

1. Generación de tráfico.
2. Distribución de tráfico.
3. Selección de modo de transporte.
4. Asignación de tráfico.

Este proceso, aunque en apariencia simple, resulta complejo y, en muchas ocasiones, se convierte en un proceso iterativo donde pudiera hacerse necesario regresar a pasos anteriores [67]. Llevar a cabo este proceso conlleva, en forma resumida, al siguiente procedimiento [67]: primero el área de estudio es dividida en varias zonas. Esta división de zonas es generalmente obtenida de los patrones de uso de la tierra (residencial, comercial, industrial) del área. Seguidamente, se estiman por cada zona una cantidad de viajes generados usando algún modelo de generación de viajes. La salida de los modelos de generación de viajes son usados para determinar el número de viajes entre pares de zonas usando algún modelo de distribución de viajes. Dado el patrón de distribución de viajes (el cual es usualmente representado como una matriz origen destino (O-D)), se estiman que porciones de viajes usarán cuál modo de viaje mediante un modelo de selección de modo. Finalmente, un modelo de asignación de viajes estimará el volumen en cada enlace de la red, determinando que ruta será usada por cada viaje.

4.4.1. Modelos de generación de tráfico

Los modelos de generación de tráfico apuntan a predecir el número de viajes generados por una zona. Hay básicamente dos modelos generales para la generación de tráfico [67]: los modelos de clasificación cruzada y los modelos de regresión. Sin embargo, ambos modelos incorporan los mismos factores básicos que afectan la generación de tráfico en una zona; ellos sólo difieren en la caracterización de estos factores.

Los factores que afectan la generación de viajes en una zona son [67]:

- El número de viajantes potenciales por zonas: los datos pueden ser capturados por variables como densidad residencial, cantidad de ocupantes por hogar, edad promedio de los ocupantes.
- La propensión de un viajante potencial de realizar un viaje: está relacionado con la tenencia de automóviles, acceso a transporte público, entre otros.

- Acceso de una zona a potenciales zonas atractoras

4.4.1.1. Modelo de clasificación cruzada

El modelo de clasificación cruzada se basa en la suposición que el número de viajes generados por hogares similares o pertenecientes a la misma categoría son los mismos [67]. De acuerdo con este modelo, si en la zona i hay n_k^i hogares en la categoría k y si g_k es el promedio de generación de viajes por hogar en la categoría k entonces la relación de los viajes generados por la zona i, T_i, es [67]:

$$T_i = \sum_{\forall k} n_k^i g_k$$

El modelo predice los viajes producidos por una zona simplemente agregando el total de viajes producidos por todos los hogares en esa zona. Sin embargo, hay dos interrogantes que responder [67]:

- ¿Cómo determinar hogares similares o en su defecto, categorías de hogares?
- ¿Cómo determinar el promedio de viajes generados por una categoría de hogar?

La respuesta a ambas preguntas es [67]: mediante observaciones y análisis empíricos. Una forma común de recoger estas observaciones es a través de encuestas realizadas a una muestra de la población en la zona estudiada [69] y/o mediante información de censo. Aún así, este método puede llegar a ser algo artesanal y posiblemente requiera de varios ajustes antes de poder producir una demanda de tráfico que se acerque a la realidad.

4.4.1.2. Modelo de regresión lineal

El método de regresión lineal asume que existe forma aditiva funcional que relaciona los factores que afectan el número de viajes generados [67]. Generalmente, la función lineal usada tiene la siguiente forma [67]:

$$T_i = \alpha_1 z_{1,i} + \alpha_2 z_{2,i} + ... + \alpha_n z_{n,i}$$

donde α_k son los parámetros de la función de regresión y $z_{k,i}$ es el valor de la $k - esima$ variable, tal como: ingreso, tenencia de automóvil, número de miembros en el hogar, etc.

Usando este modelo, determinar el número de viajes generados por una zona depende de conocer con bastante precisión el valor de los parámetros [67]. Para ello es posible determinar los parámetros con alguna técnica de estimación de parámetros [67], tal como *Mínimos cuadrados* o *Máxima verosimilitud*.

4.4.2. Modelos de distribución de tráfico

Los modelos de distribución de tráfico se aplican para determinar el número de viajes que serán realizados entre un par de zonas para un propósito de viaje determinado [67]. Ellos describen matemáticamente la fase de selección de destino en el proceso de cuatro pasos clásico. Al igual que los modelos de generación de tráfico, los modelos de distribución de tráfico incorporan los mismos factores básicos que afectan el número de viajes entre zonas, mas sin embargo, difieren en la caracterización de estos [67].

Los factores que afectan el número de viajes entre dos zonas son [67]:

- El número de viajes producidos por la zona origen.

- El grado en el cual los atributos de la zona destino atraen viajantes. Estos atributos varían de acuerdo con el propósito del viaje.

- Factores que inhiban los viajes entre un par de zonas, tales como, distancia, costo de movilización, tiempo de viaje, entre otros.

El producto de los modelos de distribución de tráfico usualmente son matrices origen destino, las cuales reflejan la cantidad de viajes entre todos los pares de zonas.

Los modelos de distribución de tráfico más usados en la práctica son los denominados modelos gravitatorios [70]. Otros modelos en uso son [67]: el modelo de entropía, el modelo de oportunidades intervinientes [3] y el modelo de selección.

4.4.2.1. Modelos Gravitatorios

Los modelos gravitatorios generalmente estiman la distribución del tráfico como una función proporcional al número de destinos producidos en una zona por los modelos de generación de tráfico e invérsamente proporcional a la suma de uno o varios factores inhibidores [69]. Este modelo ha sido muy usado debido a su simplicidad, su exactitud y el soporte que tiene de parte del Departamento de Transporte de Estados Unidos [69].

El modelo usa la siguiente forma básica para determinar los viajes realizados entre una zona origen i y una zona destino j [67]:

$$t_{ij} = K_g^i \frac{f(a_j)}{h(d_{ij})} T_i$$

donde:

- T_i es el número total de viajes producidos por la zona de origen i.

[3] "Intervening Opportunities" en inglés

- a_j es un conjunto de atributos *in-situ* de la zona destino j los cuales atraen viajes.
- $d_{i,j}$ es un conjunto de factores que inhiben viajes entre las zonas i y j.
- K_g^i es una constante de calibración.
- f y h son funciones positivas monótonamente crecientes.

La expresión $K_g^i \frac{f(a_j)}{h(d_{ij})}$ puede ser vista como el factor que distribuye el total de los viajes producidos por una zona a todas las posibles zonas destino. En este sentido, la suma de la expresión sobre todas las zonas destino debe ser igual a 1, por ende [67]:

$$\sum_{\forall j} K_g^i \frac{f(a_j)}{h(d_{ij})} = 1$$

Usar un modelo gravitatorio para generar la distribución de tráfico usualmente involucra un proceso de ensayo y error, el cual requiere especial cuidado. La calibración es usualmente llevada a cabo sobre el término K_g^i, aunque es posible que sean necesarios ajustes sobre las funciones f y h. Los ajustes son realizados incrementalmente con sucesivas iteraciones hasta lograr que la distribución de la longuitud de viaje se equipare con la distribución obtenida en las encuestas o presente una buena forma y una adecuada longitud promedio de viaje [69].

Una consideración importante en los modelos gravitatorios es balancear las producciones y las atracciones, i.e., el total de las producciones iguala el total de atracciones en el área de estudio para cada propósito de viaje [69]. Decidir si las producciones o las atracciones deben ser el total de control, depende de la confianza de los datos inherentes a cada uno de ellos, como por ejemplo, población y empleo respectivamente [69].

4.4.3. Modelos de asignación de tráfico

Los modelos de asignación de tráfico procuran determinar el número de viajes sobre diferentes enlaces pertenecientes a una red dada la demanda entre diferentes pares de nodos o zonas. Estos modelos intentan describir matemáticamente la fase de selección de ruta en el proceso de simulación.

El modelo de asignación de tráfico más simple se denomina *Modelo de asignación "todo o nada"*, el cual sigue los siguientes pasos [73]:

1. Encontrar el arbol de camino mínimo desde los nodos centroides de cada zona usando el algoritmo de Dijkstra, Floyd-Warshall o cualquier otro algoritmo eficiente de búsqueda de caminos mínimos.
2. Asignar todo el flujo correspondiente a un par de zonas al camino mínimo encontrado entre éstas.
3. Sumar cada uno de los volúmenes asignados.

Sin embargo esta solución es, por obvias razones, poco realista, puesto que no todos los conductores toman la misma vía para desplazarse entre un par de zonas. Por otro lado, esta solución conlleva a un uso ineficiente de la red de tráfico puesto que es posible que deje varias calles sin flujo.

Una opción a este modelo es el *modelo de asignación incremental* [67], el cual supone que las capacidades de los enlaces, y por consiguiente, el tiempo de viaje a través de ellos varía a medida que se van sumando vehículos a éstos. Por lo tanto, el modelo de asignación incremental trata de ajustar los valores de capacidad de cada enlace de forma iterativa, resultando en una aproximación más realista que el modelo de asignación "todo o nada".

Existen dos principios fundamentales que procuran seguir varios modelos de asignación de tráfico [71]:

1. El tiempo de viaje en todas las rutas usadas es igual o menor a aquel experimentado por un solo vehículo en cualquier ruta no usada.

2. El tiempo de viaje promedio es mínimo.

El primero de estos principios es el denominado *Equilibrio de usuario*. Este principio está fuertemente relacionado con el *Equilibrio de Nash*, (véase 3.4.2) puesto que establece que "ningún conductor podría reducir su tiempo de viaje tomando cualquier otra ruta" [71], por lo que en ocasiones el denominado *Equilibrio de Wardrop* se maneja como un juego donde intervienen muchos jugadores[75].

La forma tradicional de resolver el Equilibrio de Wardrop es mediante programación lineal puesto que es posible plantearlo como un problema convexo de optimización[75]. No obstante, para redes grandes la resolución del Equilibrio de Wardrop por programación lineal es intratable en tiempo, por lo que se hace necesario el uso de alguna otra técnica. En la sección 5.2 se discutirá sobre *Equilibrio de Usuario Dinámico* y la forma como es implementado en el framework de simulación usado para este trabajo de grado.

4.5. Modelos microscópicos basados en agentes inteligentes

Los modelos microscópicos de tráfico han existido desde hace más de 30 años, sin embargo es recientemente cuando se han hecho investigaciones para incluir el paradigma de los agentes inteligentes en los modelos microscópicos. El uso de agentes inteligentes como herramienta de simulación permite alcanzar mayor nivel de detalle en los modelos. Puesto que la tarea de conducción es considerada una tarea cognitiva y por ende como un proceso interno de un agente (en este caso, humano) [29], el paradigma de los agentes inteligentes da lugar a la representación de la percepción del conductor de su ambiente mediante sensores y la subsecuente respuesta a este, luego de un proceso de decisión a través de sus efectores. Las reglas de conducción a las que

están apegados los agentes inteligentes están regidas por algún modelo microscópico (Véase 4.3), por lo que los agentes en sí no proponen ninguna formulación sobre el movimiento longitudinal o la toma de decisiones discretas, convirtiéndose más que todo en una poderosa herramienta para simular modelos microscópicos, aprovechando las características que ofrecen los agentes inteligentes y los sistemas multiagentes.

Una característica interesante que proveen los agentes inteligentes es su capacidad para mantener y modificar a lo largo del tiempo un estado interno, lo cual bien puede ser visto como el estado psicológico o físico de conductor, o como un plan de viaje a realizar que puede variar dependiendo de las condiciones de la red de tráfico. Adicionalmente, dadas las características propias de los sistemas multiagentes, es posible modelar la heterogeneidad inherente al tráfico, tanto en los estilos de conducción como en las distintas decisiones de ruta para realizar las tareas diarias.

4.6. Software de simulación microscópica de tráfico basada en agentes inteligentes

Una definición estándar de la estructura interna de un simulador de tráfico es inexistente. Sin embargo, existe una estructura típica que los simuladores siguen frecuentemente [29]. Se considerarán dos arquitecturas básicas en el diseño de un simulador de tráfico basado en agentes inteligentes: la arquitectura para simulación basada en agentes inteligentes y la estructura del simulador de tráfico.

La primera depende fuertemente del framework de agentes sobre el cuál esté basado el simulador [42]. Aún no existe un estándar al respecto, sin embargo, se están realizando algunos desarrollos al respecto para establecer una arquitectura que se adecué a las necesidades de los sistemas multiagentes [43], [44] y [45].

En cuanto a la estructura del simulador microscópico de tráfico basado en agentes inteligentes, tampoco es posible encontrar una definición estándar, sin embargo, existen diversas propuestas de cómo debería ser la estructura de este simulador. La Figura

4.5 ejemplifica la posible estructura de un simulador microscópico de tráfico y las relaciones entre componentes.

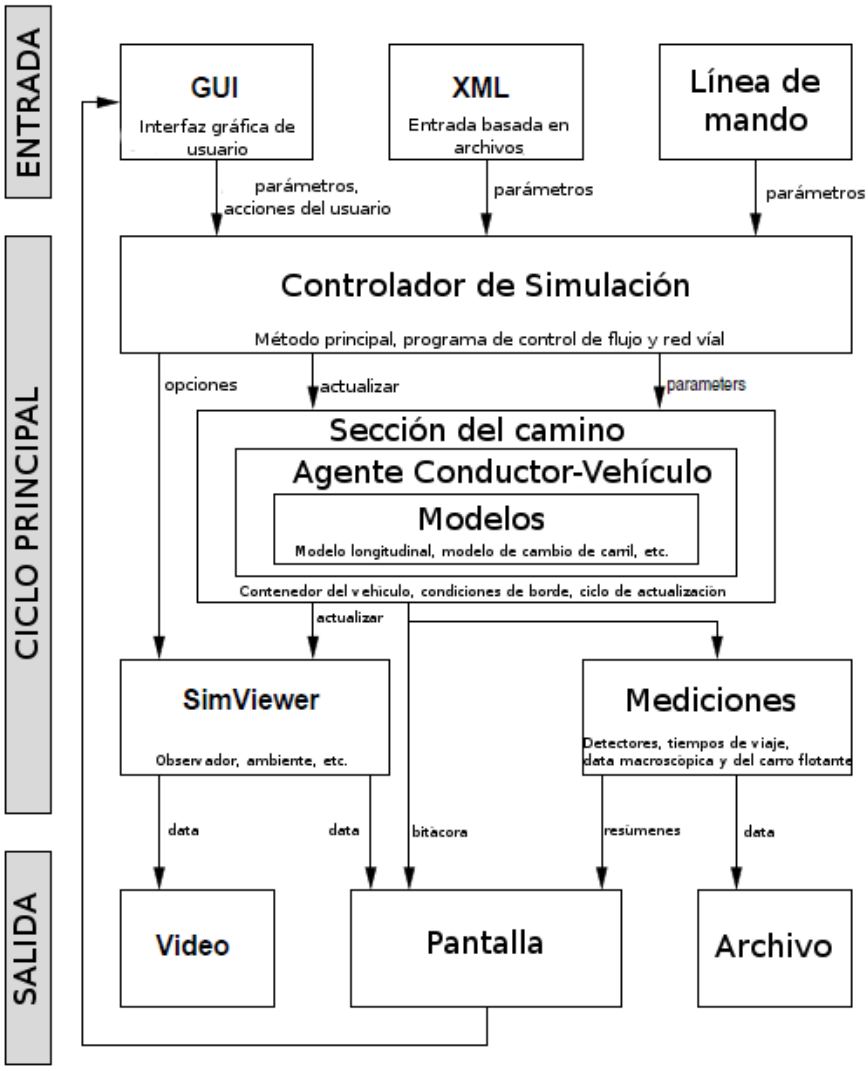


Figura 4.5: Posibles relaciones entre los componentes de un simulador de tráfico, Fuente: [29], p.344

Para la ejecución de simulaciones existen distintos simuladores microscópicos comerciales y libres a disposición del investigador, entre los cuales están:

NETSIM : Simulador software libre de calles e intersecciones [46].

FREESIM : Simulador software libre de carreteras y autopistas [47].

CORSIM : Simulador propietario que incluye NETSIM y FREESIM, permitiendo simulaciones en ambos escenarios (calles y carreteras) [48].

PARAMICS : Basado en agentes, es un software propietario de simulación microscópica de propósito general [49].

Massive Insight : Desarrollado por la compañía Massive, este software propietario permite realizar simulaciones basadas en agentes inteligentes para situaciones comunes de tránsito, tanto vehicular como peatonal [50].

SUMO : Desarrollado por el Instituto de Sistemas de Transporte del Centro Aeroespacial de Alemania, es un simulador microscópico software libre, bajo licencia GPL, basado en C++, el cual fue diseñado para el manejo de grandes redes de carreteras [51].

MATSim : Es una herramienta que permite implementar simulaciones basadas en agentes a gran escala. Desarrollado por Universidad Técnica de Berlín y por el Instituto Federal Suizo de Tecnología de Zurich es una aplicación distribuida bajo la licencia GPL que ha sido usada en diversos estudios de las carreteras de Suiza, Alemania, Indonesia, Sudafricana y Canadá [52].

Capítulo 5

MATSim: Multi-Agent Transport Simulation

> "MATSim provee un conjunto de herramientas para implementar simulaciones de tráfico a gran escala basadas en agentes inteligentes. Este conjunto de herramientas consiste en varios módulos los cuales pueden ser combinados o usados de forma independiente. Los módulos pueden ser reemplazados por implementaciones propias para probar aspectos puntuales de un trabajo. Actualmente, MATSim ofrece herramientas para el modelado de demanda, simulación de movilización basada en agentes inteligentes (simulaciones de flujo de tráfico), replanificación y gran variedad de métodos para analizar la salida generada por los módulos." [52]

Lo que diferencia MATSim de otros simuladores de tráfico tradicionales es su implementación basada en agentes inteligentes. Esto permite, entre otras cosas, describir de forma más precisa los planes de una persona durante el día, tal como ir al trabajo, a estudiar o de compras. De igual forma MATSim realiza, mediante un proceso iterativo, el proceso de asignación de tráfico gracias un método denominado *"Equlibrio dinámico de usuario"*. Otro aspecto a destacar es que MATSim permite hacer simulaciones microscópicas de gran escala gracias a su modelo de flujo vehicular basado en colas. Cada uno de estos aspectos serán descritos con detalle en este capítulo.

5.1. Planes

El tráfico no es producido por vehículos, sino por las personas que los manejan. Estas personas no producen tráfico por gusto; ellas intentan realizar sus tareas y obligaciones diarias. Es la necesidad de cumplir estas tareas y obligaciones lo que genera tráfico: la gente necesita desplazarse para poder realizarlas. Por otro lado, este desplazamiento sigue una planificación de tiempo: las personas van al trabajo, la escuela o a entretenerse en un horario específico, el cual no necesariamente es el mismo para todos.

Para simular un día típico en un área urbana, el simulador microscópico necesita información sobre la planificación de cada individuo y algún conocimiento sobre el proceso de toma de deciciones de las personas [74]. El principal reto es crear esta demanda individualizada a partir datos generales, tales como información de censo o la cantidad de puestos de trabajos en una zona.

Esta individualización de la demanda se conoce como desagregación y es un concepto fundamental en la filosofía de MATSim. Para obtener la planificación de un conjunto de personas de forma desagregada, se hace necesario seguir los siguientes pasos [75]:

1. Generar una *población sintética* mediante un muestreo aleatorio de los datos de censo sobre la región estudiada. Esto significa crear un conjunto virtual de personas que habitan la región. Este conjunto, aún cuando no tiene una relación exacta con la población real en la zona, mantiene la misma estructura demográfica.

   Una población sintética típica está compuesta por hogares los cuales poseen ciertos atributos, tales como dirección, nivel de ingreso o tenencia de vehículo. Estos hogares están a su vez, habitados por individuos que poseen atributos adicionales tales como edad o sexo.

2. Generar, por cada individuo de la población sintética, un *programa completo de actividades diarias*. Las actividades se refieren a acciones tales como "estar en casa", "estar en la escuela", "trabajar", "ir de compras", etc. Cada programa de actividades de un individuo contiene el patrón de actividades que éste desea realizar junto con el lugar donde realizará la actividad. Igualmente, contiene información temporal, tal como cuando comenzará y terminará la actividad, o cuanto durará.

3. Escoger por cada individuo de la población sintética un *modo* de transporte para cada viaje realizado entre un par de actividades.

Sin embargo, no siempre es posible obtener con detalle esta información, por lo que MATSim también permite el uso de información agregada, por ejemplo, matrices origen destino, para realizar simulaciones sobre una zona.

En MATSim, la información relativa a los viajes de cada persona en la red es recogida en forma de *planes*. Un plan contiene la planificación diaria de una persona. Cada persona es tratada en MATSim como un agente inteligente. Un ejemplo de un plan diario es:

- Trabajar de 8 AM a 4 PM.
- Ir de compras de 5 PM a 6 PM.
- Regresar a casa.
- Salir al cine de 8 PM a 10 PM.
- Volver a casa a dormir.

que en MATSim se representa en formato XML de la siguiente forma:

```
1 <person id="1">
2   <plan type="car">
3     <act type="h" x="-200" y="0" link="1" />
4     <leg mode="car">
5       <route>2 7 12 2</route>
6     </leg>
7     <act type="w" x="100" y="0" link="20" start_time="08:00" \linebreak
          end_time="16:00" />
8     <leg mode="car">
9       <route>13 14 15</route>
10    </leg>
11    <act type="s" x="-500" y="13" link="30" start_time="17:00" \
          linebreak end_time="18:00"/>
12    <leg mode="car">
13      <route>20 17</route>
14    </leg>
15    <act type="h" x="-200" y="0" link="1"/>
16    <leg mode="car">
17      <route>10 7</route>
18    </leg>
19    <act type="l" x="-200" y="330" link="8" start_time="20:00" \
          linebreak end_time="22:00"/>
20    <leg mode="car">
21      <route>7 10</route>
22    </leg>
23    <act type="h" x="-200" y="0" link="1"/>
24  </plan>
25 </person>
```

Observese que un plan también incluye la ruta que tomara el agente para movilizarse de una actividad a otra. Esta información puede ser opcional cuando se genera el plan inicial, puesto que el proceso de asignación de tráfico de MATSim es capaz de

generar cada una de las rutas iniciales para el agente en cuestión.

5.2. Asignación de tráfico mediante equilibrio dinámico de usuario

La asignación dinámica de tráfico se ha vuelto en los últimos 20 años en una técnica que pretende sobrepasar las limitaciones de la asignación estática de tráfico y producir volúmenes de tráfico dependientes del tiempo sobre los enlaces de la red[75].

Como se mencionó en la subsección 4.4.3, existen modelos analíticos para resolver el equilibrio de usuario, incluso en el caso dinámico, donde el factor tiempo asume un papel fundamental. El algoritmo para resolver el equilibrio dinámico de usuario propuesto por Ran y Boyce [76] requiere programación lineal con $O(MNT)$ variables, donde M es el número de enlaces en la red, N es el número pares origen destino y T es el número de pasos de tiempo a ser resueltos en una iteración. Para grandes redes, con miles de nodos y pares origen destino, esta solución resulta impráctica.

Una forma pragmática de encontrar el equilibrio dinámico de usuario es mediante *simulación iterativa*, la cual consiste en [66] [75]:

1. Escoger las rutas iniciales suponiendo tráfico cero.
2. Calcular la carga en la red y los tiempos de viaje mediante simulación, actualizando seguidamente las rutas de los conductores.
3. Volver a realizar el proceso, modificando algunas rutas, hasta que los tiempos de viaje se vuelvan estacionarios.

MATSim utiliza este enfoque para resolver el problema de la asignación de tráfico, denominándolo *Relajación sistemática*[75]. Esta relajación sistemática utiliza un algoritmo evolutivo que realiza los siguientes cambios sobre los planes de los agentes[75]:

- Cambio de ruta.
- Cambio de hora de inicio de una actividad.
- Cambio de la función de utilidad.
- Cambio de modo.
- Usar el mejor plan conseguido.

Cada uno de los cambios anteriores tiene una probabilidad asociada fijada antes del inicio del proceso de relajación. Cada agente posee además memoria, recordando un conjunto de planes ejecutados con antelación, además del mejor plan encontrado hasta el momento. Los planes son puntuados usando una función de utilidad basada en el costo del viaje. Esta función de utilidad tiene la siguiente forma [75]:

$$U_{total} = \sum_{i=1}^{n} U_{realizar,i} + \sum_{i=1}^{n} U_{tarde,i} + \sum_{i=1}^{n} U_{viajar,i}$$

donde U_{total} es la utilidad total de un plan dado; n es el número de actividades; $U_{realizar,i}$ es la utilidad positiva asociada a realizar la actividad i, $U_{tarde,i}$ es la utilidad negativa asociada a llegar tarde a la actividad i; y $U_{viajar,i}$ es la utilidad negativa asociada a viajar hasta la actividad i. A efectos de trabajar con unidades del mundo real, las utilidades se miden en dinero [75].

Para que este enfoque pueda funcionar en términos de tiempo, se hace necesario un modelo de simulación de flujo vehicular que sea simple y rápido. El modelo estocástico de colas implementado en MATSim resuelve este problema.

5.3. Modelo estocástico de cola para simulación de tráfico basada en agentes inteligentes

El modelo de flujo vehicular implementado en MATSim es el propuesto por Gawron (véase 4.3.2) con algunas modificaciones. En el modelo original, la selección del

siguiente enlace cuando un vehículo o un conjunto de ellos se aproxima al final de su enlace se realiza de forma arbitraria pero siempre en el mismo orden. Eso quiere decir, que ciertos enlaces tienen preferencia a la hora de ser procesados [65].

Es posible modificar el algoritmo de forma que se priorice la selección del enlace de forma aleatoria proporcional a la capacidad de éste. Eso significa que los enlaces con mayor capacidad tienen mayor probabilidad de ser seleccionados que aquellos con menos capacidad [68]. En MATSim, el algoritmo original, que está orientado a los enlaces, fue modificado para que esté orientado a las intersecciones, lo cual se traduce en que por cada iteración se revisarán las intersecciones y de ellas, los enlaces incidentes [68]. Adicionalmente, se separa la restricción de capacidad de flujo de la lógica de la intersección. Para lograr esto, se introduce un *buffer* separado al final del enlace, cuyo tamaño es igual a la capacidad del enlace. Los vehículos son movidos del enlace al *buffer* de acuerdo con la restricción de capacidad y si hay suficiente espacio en el *buffer*; una vez en el *buffer* los vehículos pueden ser movidos a través de la intersección al siguiente enlace si tiene espacio [65].

A continuación se muestran los algoritmos, tanto del modelo original como del modelo modificado[68].

Algoritmo del modelo original de Gawron:

```
PARA cada enlace HACER
  MIENTRAS un vehiculo ha llegado al final del enlace
  Y el vehiculo puede ser movido de acuerdo a la capacidad
  Y hay espacio en el enlace destino HACER
    Mover el vehiculo al siguiente enlace
  FIN MIENTRAS
FIN PARA
```

Algoritmo modificado:

```
// Propagar los vehiculos a lo largo de los enlaces
```

```
PARA cada enlace HACER
  MIENTRAS un vehiculo haya llegado al final del enlace
  Y el vehiculo puede ser movido de acuerdo a la capacidad
  Y hay espacio en el buffer HACER
    Mover el vehiculo del enlace al buffer
  FIN MIENTRAS
FIN PARA
// Mover los vehiulos a traves de las intersecciones
PARA cada interseccion HACER
  MIENTRAS existan enlaces elegibles HACER
    Seleccionar un enlace aleatoriamente de acuerdo a su capacidad
    Marcar este enlace como no elegible
    MIENTRAS existan vehiculos en el buffer de ese enlace HACER
      Verificar el primer vehiculo en el buffer del enlace
      SI el enlace destino tiene espacio ENTONCES
        Mover el vehiculo desde el buffer al enlace destino
      FIN SI
    FIN MIENTRAS
  FIN MIENTRAS
FIN PARA
```

Una consecuencia de esta modificación es que permite que el algoritmo de actualización sea completamente paralelizable: si el tráfico es movido completamente fuera de un enlace, el nuevo espacio sólo se abrirá en el *buffer* y no en el enlace, por ende este espacio no estará disponible en la siguiente intersección sino hasta el siguiente paso de tiempo [68]. Esto tiene la ventaja que toda la información que es necesaria en el cómputo de un paso de tiempo está disponible localmente en cada intersección antes que el paso de tiempo comience, por lo tanto, no hay intercambio de información entre las intersecciones durante el cómputo del paso de tiempo [65].

5.4. ¿Cómo usar MATSim?

Existen dos tipos de usarios en MATSim: usuarios del *toolkit* y usuarios del *API*. Los primeros, generalmente usarán las clases contenidas en el paquete *run*, el cual contiene las clases ejecutables para realizar la mayoría de las tareas elementales a la hora de preparar el escenario de simulación, efectuar la simulación y analizar los resultados. Por otro lado, los usuarios de API son aquellos que utilizan el API de MATSim para realizar simulaciones personalizadas que requieran alguna modificación menor sobre el comportamiento del simulador.

Finalmente, existe otra categoría de usuarios, mucho más avanzados: los desarrolladores. En esta categoría caen las personas que intervienen creando nuevos componentes de software o modificando los componentes del *core* del simulador para fines investigativos. Esta sección se enfocará sobre cómo usar el *toolkit* de simulación.

El paquete org.matsim.run contiene las clases ejecutables del *toolkit*:

Benchmark : ejecuta una simulación de prueba usando como escenario a Berlín.

Controler : ejecuta una simulación usando como escenario aquel que sea indicado en el archivo de configuración pasado por argumento.

CreateSelectedPlansTables : genera estadísticas de cada uno de los agentes a partir de los planes: tiempo de viaje, distancia recorrida y número de viajes.

Events2Snapshots : convierte un archivo de eventos a un archivo de *snapshots* de forma que estos eventos puedan ser visualizados en OTFVIS.

InitRoutes : crea rutas iniciales en un archivo de planes cuyas rutas no hayan sido establecidas.

NetworkCleaner : limpia la red, asegurando que es apta para la simulación. Por ahora, su trabajo consiste en hacer que la red sea conexa.

OTFVis : permite visualizar tanto redes como simulaciones.

RealeaseInfo : imprime información del paquete.

TeleAtlas2Network : convierte redes TeleAtlas a redes MATSim.

XY2Links : Dado un archivo de planes cuyas actividades están localizadas solamente en posiciones (x, y), genera un archivo de planes donde todas las actividades tienen un enlace relacionado.

Para realizar una simulación en MATSim son necesarios 3 elementos:

- Archivo de configuración: este archivo XML contiene información sobre cómo debe ser realizada la simulación (semilla aleatoria, sistema de coordenadas a usar, parámetros de las funciones de cálculo de utilidad en los planes, entre otras cosas), además de indicar la localización del archivo XML que contiene los planes y el archivo XML que contiene la red. Véase apéndice A.
- Planes: un archivo XML conteniendo los planes de los agentes. Véase 5.1 y apéndice B.
- Red: un archivo XML que almacena la estructura de la red. Las redes son representadas como grafos, donde las intersecciones son vértices y las aristas son calles. Véase apéndice C.

Adicionalmente, se puede incluir un archivo XML que contenga la información de los conteos de tráfico para realizar validaciones sobre la simulación.

Una vez que se tienen estos tres elementos, es posible realizar simulaciones ejecutando la clase *Controler*. Este programa generá un conjunto de salidas colocadas en carpetas separadas por cada iteración. La salida de la simulación está compuesta por:

- Planes generados por la iteración, los cuales poseen las rutas optimizadas de todos los agentes.
- Archivos de eventos y *snapshots* para visualizar la simulación.
- Si se incluyeron conteos, se generarán archivos que indicarán la diferencia entre los volúmenes de tráfico observados y los obtenidos en la simulación.
- Histogramas relativos a los eventos sucedidos durante la simulación: número de vehículos circulando, número de salidas de una actividad y número de llegadas a una actividad.
- Duración de los viajes.
- Flujo asignado a cada enlace por hora.
- Distancias viajadas: promedio y máximo.

En los apéndices G y H se puden encontrar ejemplos de gráficos generados por el programa *Controler*.

Capítulo 6

Herramientas y entorno de desarrollo

En este capítulo se explican cada uno de los recursos necesarios para la consecución de los objetivos planteados en 1.4

6.1. Lenguaje de programación

Debido a que MATSim está enteramente implementado en Java, la escogencia del lenguaje de programación está atado a ello. Las primeras versiones de MATSim fueron implementadas en C++, sin embargo el desarrollo de MATSim fue migrado a Java paulatinamente. En palabras de Marcel Rieser, desarrollador del proyecto MATSim:

> "Algunas funcionalidades fueron primero probadas e implementadas por estudiantes de pre-grado para sus proyectos de tesis. Con el tiempo, más y más estudiantes trabajando en MATSim poseían gran conocimiento de ingeniería de transporte, pero poco conocimiento en ciencias de la computación. El manejo manual de la memoria en C++ conllevaba a inestabilidades del sistema debido a fugas de memoria, memoria sobrescrita innadecuadamente, entre otros. Java es mucho más robusto en este aspecto, y nos permitió agregar nuevas funcionalidades más rápidamente, a la vez que se obtenía gran estabilidad en el sistema. Como efecto colateral positivo, se obtuvo un sistema independiente de plataforma."

6.2. API

El API utilizado proviene del *framework* de desarrollo de MATSim, el cual es descrito en el capítulo 5 y los apéndices D, E y F. De igual forma, se usa API provisto por el conjunto de clases contenidas en el paquete *org.geotools* las cuales facilitan la manipulación de archivos SHP y los datos contenidos en éstos.

6.3. Gestión de proyecto: Maven

Maven es una herramienta de software para la administración y construcción de proyectos. Aún cuando es usado primordialmente para proyectos en Java, puede ser usado también para construir y administrar proyectos escritos en C#, Ruby y Scala, entre otros. Maven sirve para un propósito similar a la herramienta *Apache Ant*, pero está basado en conceptos diferentes y trabaja de forma distinta. Inicialmente, Maven formó parte del proyecto Jakarta, pero ahora ya es un proyecto de nivel superior de la *Apache Software Foundation*.

Maven usa una construcción denominada *Project Object Model* (POM) para describir el proyecto de software a construir, sus dependencias y otros módulos y componentes externos, y su orden de construcción. Maven posee objetivos predefinidos para realizar tareas determinadas, tales como compilación, depuración o distribución.

Maven dinámicamente descarga las bibliotecas y los *plug-ins* de Java necesarios para un proyecto de uno o varios repositorios. Maven provee soporte nativo para obtener archivos del repositorio central de Maven (*Maven 2 Central Repository*) y de otros repositorios Maven, permitiendo además subir artefactos específicos luego de realizar una construcción exitosa. Un caché local de artefactos actúa como el primer medio para lograr la sincronización de los proyectos producidos localmente.

6.4. Control de versiones

El desarrollo en equipo requiere de algún método que permita integrar los desarrollos individuales de forma automática y organizada. Para tal fin, se ha usado *Mercurial* como software para el soporte del control de versiones. En la página web de *Mercurial* es posible encontrar la siguiente descipción [79]:

> Mercurial es una herramienta de control de versiones gratuita y distribuida. Ésta permite manipular proyectos de cualquier tamaño a la vez que se usa una interfaz intuitiva. Es fácil de usar y difícil de dañar, haciéndola ideal para cualquiera que trabaje con archivos versionados.

El repositorio de desarrollo está almacenado en *bitbucket*, el cual es un servicio de hosting web desarrollado en *Python* que usa el sistema de control de versiones de *Mercurial*.

Capítulo 7

Visualizador de redes de tráfico: diseño, implementación y pruebas

7.1. Diseño del visualizador de la red

Un elemento que no se encuentra implementado en el *framework* de MATSim es un programa que permita la edición de una red en formato nativo de MATSim. Aún cuando existe un visualizador, éste no provee funcionalidades para la edición de redes de tráfico, al igual que funcionalidades que permitan la conversión entre formatos de redes distintos al de MATSim.

El objetivo de la implementación del visualizador consiste en facilitarle el trabajo al analista de tráfico a la hora de afinar detalles en una red de tráfico determinada, permitiendo no sólo modificar los atributos de los enlaces, sino también agregar nuevos enlaces y eliminar aquellos que no sean necesarios. El visualizador igualmente permite la inclusión de estaciones de conteo en la red y la modificación de los atributos de éstas. Por otro lado, el visualizador facilita el trabajo de conversión entre el formato OSM [77] (*Open Street Maps*) y el formato nativo de MATSim, al igual que hace posible exportar una red MATSim a formato SHP [78].

7.1.1. Requisitos funcionales del visualizador

Como se vio en la sección 7.1, el visualizador debe facilitar varias tareas de manipulación de la red al analista de tráfico. Para resolver tales requisitos se necesita específicamente de las siguientes funcionalidades:

1. Cargar una red a partir de un archivo en formato OSM.
2. Cargar y guardar una red MATSim.
3. Exportar una red a formato SHP.
4. Cargar y salvar conteos en formato MATSim.
5. Acercar y alejar una zona de la red así como moverse por la red.
6. Seleccionar una intersección.
7. Seleccionar un enlace o un conjunto de enlaces.
8. Visualizar la dirección y la capacidad de los enlaces en la red.
9. Crear nuevos enlaces.
10. Eliminar algún(os) enlace(s) seleccionado(s).
11. Agregar una estación de conteo al enlace seleccionado.
12. Agregar, eliminar y modificar conteos de una estación de conteo.
13. Visualizar los atributos del elemento seleccionado, siendo posible la modificación de éstos.
14. Dehacer y rehacer cambios hechos sobre la red.

En las siguientes subsecciones se revisarán las clases destinadas a solucionar los anteriores requisitos y las funciones definidas dentro de ellas que los resuelven.

Para entrar en detalles del diseño del visualizador es preciso repasar la estructura que representa una red en MATSim: la interfaz Network.

7.1.2. La interfaz *Network* en MATSim

Una red de tráfico en MATSim es representada mediante un grafo dirigido, donde las intersecciones son los vértices y las calles aristas. La interfaz *Network* en MATSim está contenida en el paquete *org.matsim.api.core.v01.network*. Véase el apéndice D para detalles sobre las funciones definidas en la interfaz.

Puesto que *Network* es una interfaz, es necesario una implementación para poder instanciar un objeto del tipo *Network*. En MATSim, esta implementación está en la clase *NetworkImpl*, definida en el paquete *org.matsim.core.network*. Esta clase no sólo implementa las funciones definidas en la interfaz *Network*, sino que también define e implementa nuevos atributos y funciones. De particular interes es la inclusión de un *Quad Tree* o árbol cuaternario para almacenar los nodos. Este árbol permite la búsqueda rápida de nodos y enlaces en un rango determinado, haciendo posible la fácil implementación de importantes funcionalidades en el visualizador, como por ejemplo, la selección de un nodo o un enlace al hacer *click* sobre él.

Node y *Link* son de igual manera interfaces, definidas en el mismo paquete donde se encuentra la interfaz *Network*. Estas interfaces proveen un conjunto de funciones para acceder y modificar la información de intersecciones y enlaces en la red de tráfico MATSim.

7.1.3. La interfaz *Node* y *Link* en MATSim

La interfaz *Node* hereda de dos interfaces: *BasicLocation* y *Identifiable*; permitiendo que los nodos posean tanto una localización en forma de coordenadas y un

identificador único. Véase el apéndice E para detalles sobre las funciones definidas en la interfaz.

La interfaz *Link* hereda de la interfaz *BasicLocation*. Véase el apéndice F para detalles sobre las funciones definidas en la interfaz.

La implementación de ambas interfaces, las clases *NodeImpl* y *LinkImpl* respectivamente, se encuentra en el paquete *org.matsim.core.network*.

7.1.4. Visión global del visualizador

El visualizador está compuesto principalmente por 5 clases, las cuales son a la vez componentes gráficos dentro de la aplicación. En la figura 7.1 se puede apreciar estas clases y la relación existente entre ellas.

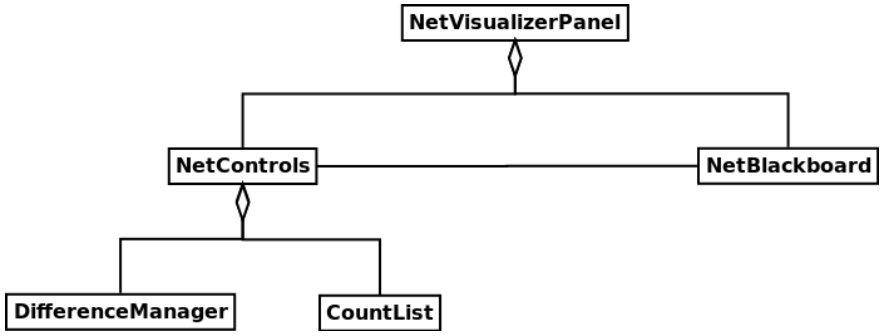


Figura 7.1: Visión general del visualizador

Las clases *NetVisualizerPanel*, *NetBlackboard* y *NetControls* heredan de la clase *JPanel* y son componentes gráficos en el visualizador. Debido a las funcionalidades provistas por las clases *NetBlackboard* y *NetControls* ambas clases están áltamente acopladas, por lo que es necesario integrar ambas a la interfaz de usuario a la hora de hacer cualquier labor con el visualizador.

7.1.5. La clase NetVisualizerPanel

La clase *NetVisualizerPanel* integra visualmente las clases *NetBlackboard* y *NetControls*. Esta clase contiene funciones que permiten cargar redes desde los formatos MATSim y OSM, exportar redes a formato SHP y cargar conteos en formato MATSim, las cuales resuelven los requisitos 1, 2, 3 y 4 expuestos en la sección 7.1.1.

La clase *NetVisualizerPanel* define las siguientes funciones:

loadNetFromFile(String Path) : carga una red en formato MATSim.

loadNetFromOSM(String Path) : carga una red en formato OSM y la convierte a formato MATSim.

loadCountsFromFile(String Path) : carga un archivo de conteos en formato MATSim.

saveNetwork(String Path) : guarda una red en formato MATSim en un archivo.

saveNetworkAsESRI(String Path) : convierte una red en formato MATSim a formato SHP y la guarde en un archivo.

7.1.6. La clase NetBlackboard

Esta clase resuelve los requisitos del 5 al 12 expuestos en 7.1.1 a través de las siguientes funciones:

zoom(double qt) : Acerca o aleja la vista de la red.

moveViewBox(double dx, double dy) : mueve la vista de la red.

setActiveSomething(Coord Pos) : obtiene el enlace o la intersección más cercana a una posición de la pantalla y lo establece como elemento activo.

paintComponent(Graphics g) : despliega por pantalla la red.

createLink(LineOnBoard line) : crea un nuevo enlace a partir de una línea dibujada en pantalla.

deleteActiveLinks() : elimina el(los) enlace(s) seleccionado(s).

addCountingStation(Link link, String stationName) : agrega una estación de conteo en el enlace indicado por el parámetro *link* asignándole el nombre *stationName*.

addCount(Link link, int h, double volumen) : agrega un nuevo conteo a la estación de conteo asociada al enlace *link*.

editCount(Link link, int h, double volumen) : modifica el conteo en la estación de conteo asociada al enlace *link* y cuya hora sea la indicada por el parámetro *h*.

deleteCount(Link link, int h) : elimina el conteo en la estación de conteo asociada al enlace *link* en la hora indicada por el parámetro *h*.

7.1.7. La clase NetControls

Esta clase resuelve los requsitos 11 y 12 expuestos en la sección 7.1.1. El requsito 11 es resuelto mediante objetos del tipo *JTable* y *JTableModel* insertados en el panel. El requisito 12 es resuelto a través de la clase *DifferenceManager*, la cual es usada en la clase *NetControls*.

Por otro lado, la clase *NetControls* sirve de interfaz al usuario para realizar las funciones asociadas a los requsitos del 5 al 10.

7.1.7.1. La clase DifferenceManager

Esta clase resuelve el requisito 12 del visualizador. Para ello se definen las siguientes funciones:

saveState(...) : salva un estado de la red en el historial de acciones.

undo() : deshace una acción realizada sobre la red.

redo() : rehace una acción realizada sobre la red.

Por otro lado, las modificacionese permitidas sobre la red son:

- Modificar los atributos de un enlace.
- Agregar un enlace nuevo.
- Eliminar un enlace existente.

7.1.7.2. La clase CountList

Esta clase resuelve, desde el punto de vista visual y de interacción con el usuario, las tareas de manipulación de las estaciones de conteo y los conteos asociados a ellas. A través de este panel es posible:

- Crear una estación de conteo asociada a un enlace.
- Cambiar el nombre de una estación de conteo.
- Agregar, eliminar y editar conteos asociados a una estación de conteo determinada.

7.2. Implementación del visualizador

7.2.1. Despliege de la red

El visualizador puede ser visto como una caja colocada encima de la red, la cual puede ser movida, recortada o ampliada para mover, acercar y alejar la vista de la red. Esta caja está definida por 3 variables:

1. Coordenada (x, y) inferior izquierda que delimita la caja.
2. *Offset* en x.
3. *Offset* en y.

Estas tres variables permiten concer qué parte de la red se está visualizando. Sin embargo, puesto que cabe la posibilidad de manejar redes de tamaño considerable, como zonas de una ciudad o la ciudad completa, es necesario seleccionar un conjunto de nodos y enlaces reducido tal que sean los que se observan a través de la caja. Para ello, se hizo uso de la función *getNearestNodes(Coord c, double dist)* implementada en la clase *NetworkImpl*. Dado que esta función busca nodos en una región circular, es necesario que el parámetro de distancia sea ligeramente mayor que la distancia que cubre la caja de visualización, de tal forma que sean tomados todos los enlaces que se encuentran dentro de la caja.

7.2.2. Manipulación de la red

El programa de visualización permite dos operaciones sobre la red:

- Crear nuevos enlaces
- Eliminar enlaces

La función para dibujar nuevos enlaces requiere una clase contenida en la clase *NetBlackboard* llamada *LineOnBoard* la cual contiene información relacionada con la línea que se está dibujando y los nodos/enlaces que une. Al momento de dibujar un enlace, éste se *adhiere* a un enlace o a un nodo, escogiendo lo más cercano al punto de inicio o fin. Para lograr esto, se hace una llamada a la fución *getClosestThing(Coord Pos)*, la cual devuelve un objeto de la clase *PairLinkNode* (contenida en la clase *NetBlackboard*). Esta función trabaja buscando los enlaces y los nodos más cercanos a la coordenada suministrada por parámetro, midiendo la distancia a éstos y asociando la coordenada con el más cercano de ellos. En caso que la distancia al nodo o enlace más cercano sobrepase una tolerancia calculada en base al tamaño de la caja de visualización, la función retorna un par nulo indicando que la coordenada no puede ser adherida a ningún nodo o enlace. En caso de ser escogido un enlace, se crea un nodo nuevo que conecta el nuevo enlace con el escogido.

La eliminación se realiza a través del API de MATSim, con la única particularidad que en caso de eliminar un enlace, si alguno de los nodos que conforman sus extremos queda sin ninguna otra conexión, éste es eliminado también de la red.

7.2.3. Manejo de diferencias: funciones *deshacer* y *rehacer*

Como se vió en la sección 7.1.7.1, la class *DifferenceManager* soluciona los asuntos relacionados con las funciones de revertir y rehacer acciones sobre la red. En ella se definen tres tipos de modificaciones posibles sobre la red, más específicamente, sobre los enlaces:

- Creación
- Eliminación
- Edición

Ésta clase contiene una clase llamada *Difference* que gestiona las operaciones relacionadas con la red, para lo cual se definen dos funciones públicas:

- revert(): revierte una operación hecha sobre la red, es decir, la asociada a la funcionalidad *deshacer*.
- apply(): aplica una modificación sobre la red, implementando la funcionalidad de *rehacer*.

Para almacenar los estados de la red, se usan dos pilas: una para las operaciones de *deshacer* y otra para las operaciones de *rehacer*.

7.3. Pruebas realizadas sobre el visualizador

Para las pruebas a realizarse en la presente sección se usará la red que modela la zona de Los Cortijos y Los Ruices, siendo esta la que además se usará en las simulaciones de prueba. Se escogieron 5 pruebas a partir de los requisitos funcionales descritos en 7.1.1 en base a su utilidad y/o a la dificultad de su implementación.

7.3.1. Cargar una red a partir de un archivo en formato OSM

La imagen 7.2 representa la red de los cortijos vista desde el visualizador de mapas OSM disponible en [77].

Figura 7.2: Red de Los Cortijos de Lourdes y Los Ruices en formato OSM

Luego de cargarla con el visualizador, se obtiene el resultado apreciable en la imagen 7.3.

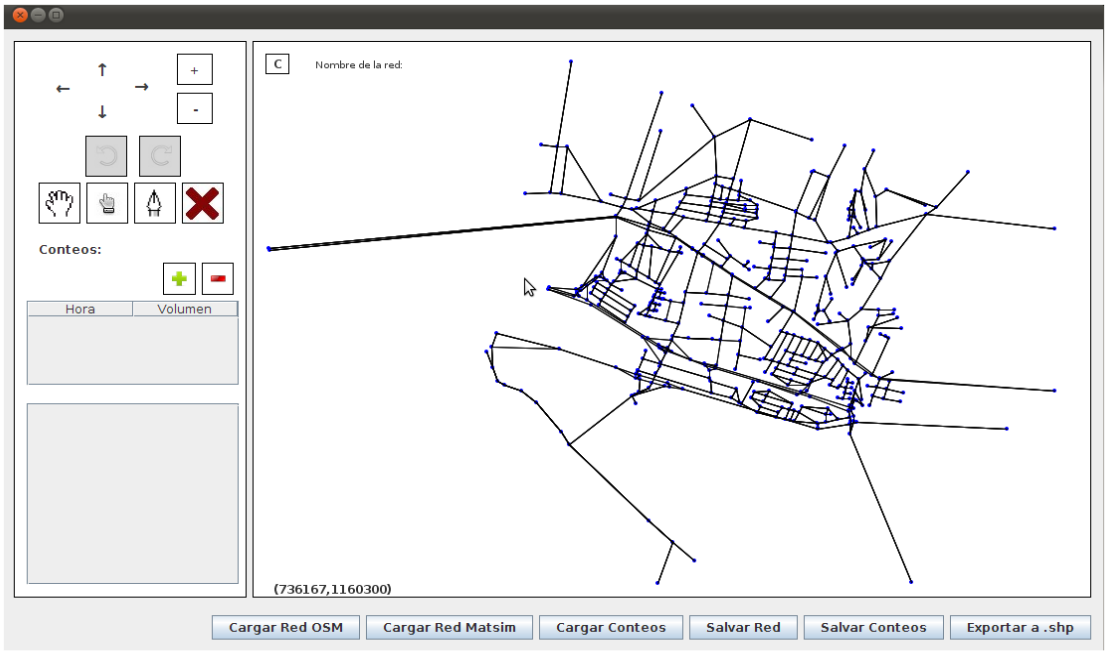


Figura 7.3: Red de Los Cortijos de Lourdes y Los Ruices cargada desde un archivo en formato OSM

7.3.2. Exportar una red a formato SHP

Tomando la red que se muestra en la imagen 7.3, se procedió a convertirla a formato SHP, obteniendo como resultado un archivo SHP mostrado en la imagen 7.4.

Figura 7.4: Red de Los Cortijos de Lourdes y Los Ruices convertida a formato SHP

7.3.3. Crear nuevos enlaces

Se procederá a crear un nuevo enlace entre una intersección y un enlace, produciendo de esta forma un nuevo nodo en la red que represente la intersección entre el enlace destino y el nuevo enlace. En la imagen 7.5 se puede apreciar un acercamiento a la red de Los Cortijos de Lourdes y Los Ruices donde se realizará la creación del nuevo enlace.

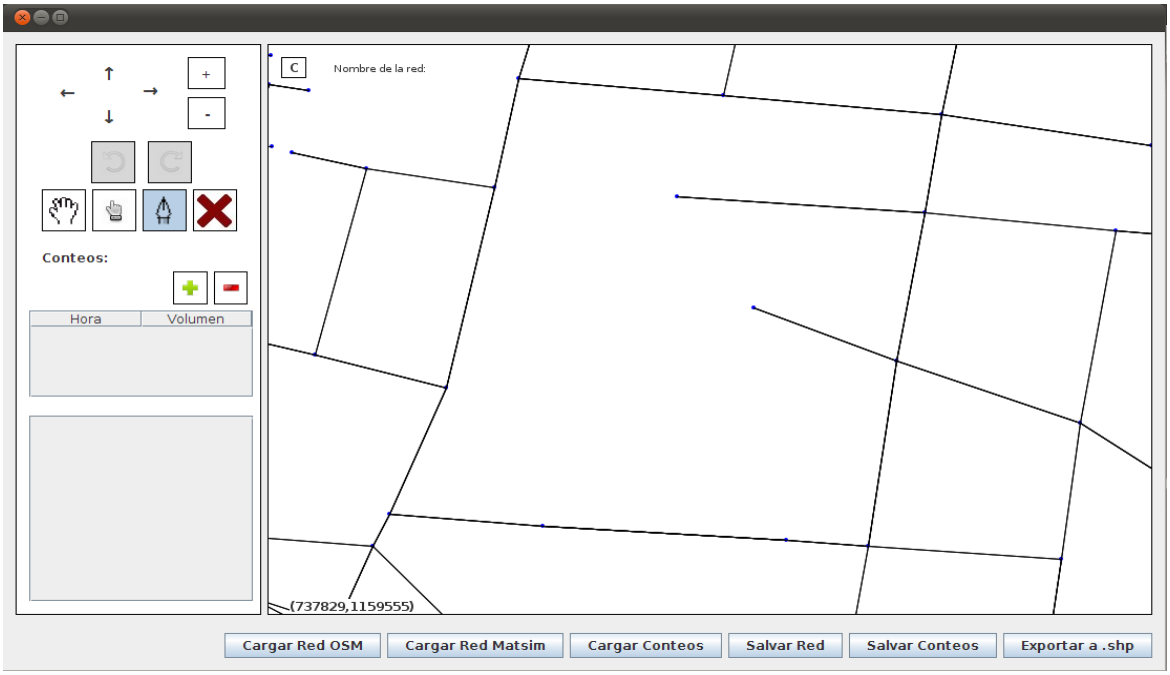


Figura 7.5: Acercamiento de la red de Los Cortijos de Lourdes y Los Ruices

En la imagen 7.6 se puede apreciar el nuevo enlace creado, así como también la dirección de este.

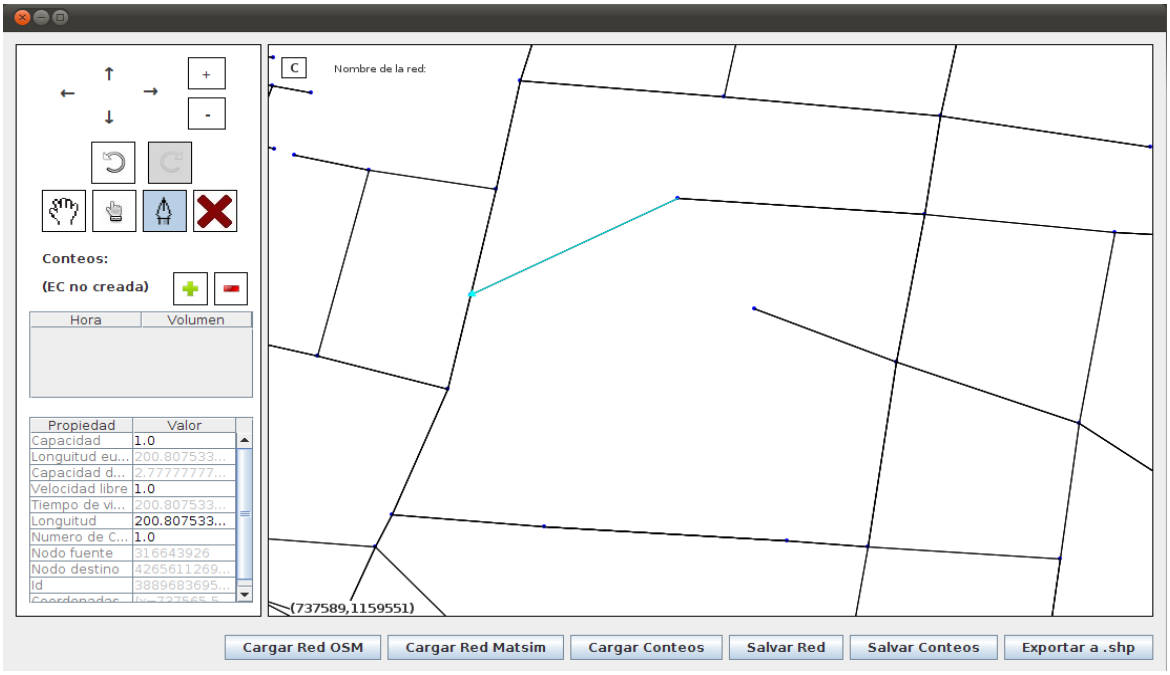


Figura 7.6: Nuevo enlace creado entre la Segunda Transversal y la Avenida Don Diego de Cisneros

7.3.4. Agregar una estación de conteo al enlace seleccionado

A partir del enlace creado en la sección 7.3.3, se procederá a agregar una estación de conteo llamada *Estación de prueba* en este enlace y un conteo a las 8 AM. Las figuras 7.7 y 7.8 muestran gráficamente las operaciones realizadas.

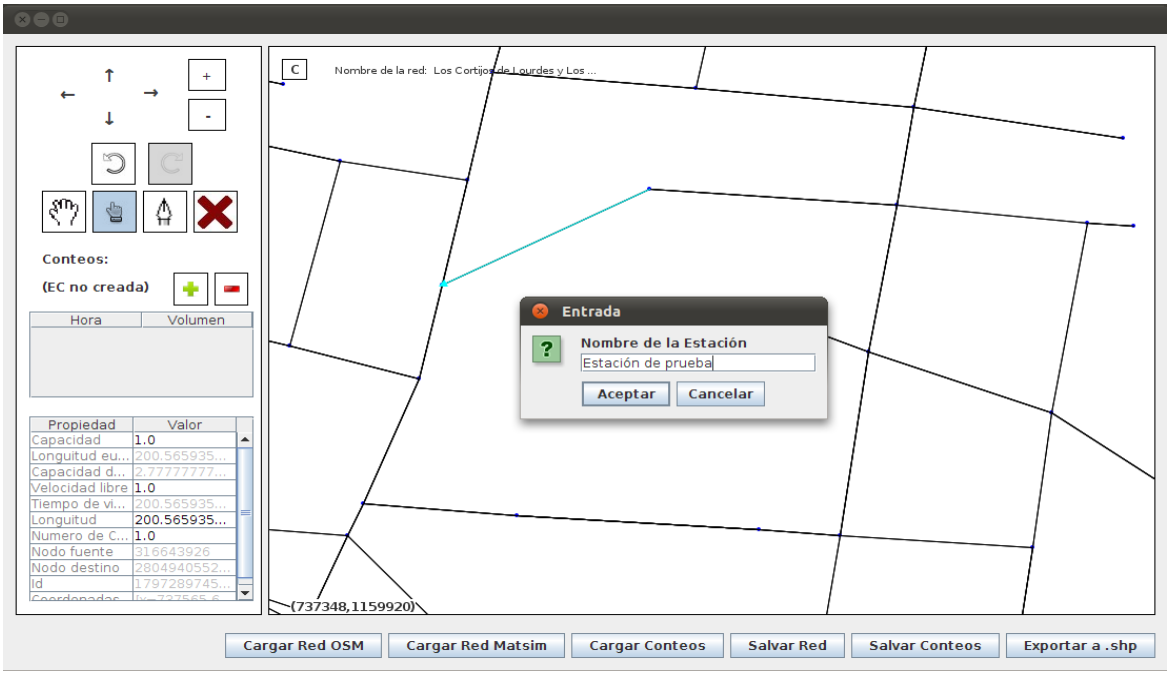


Figura 7.7: Creación de una nueva estación de conteo

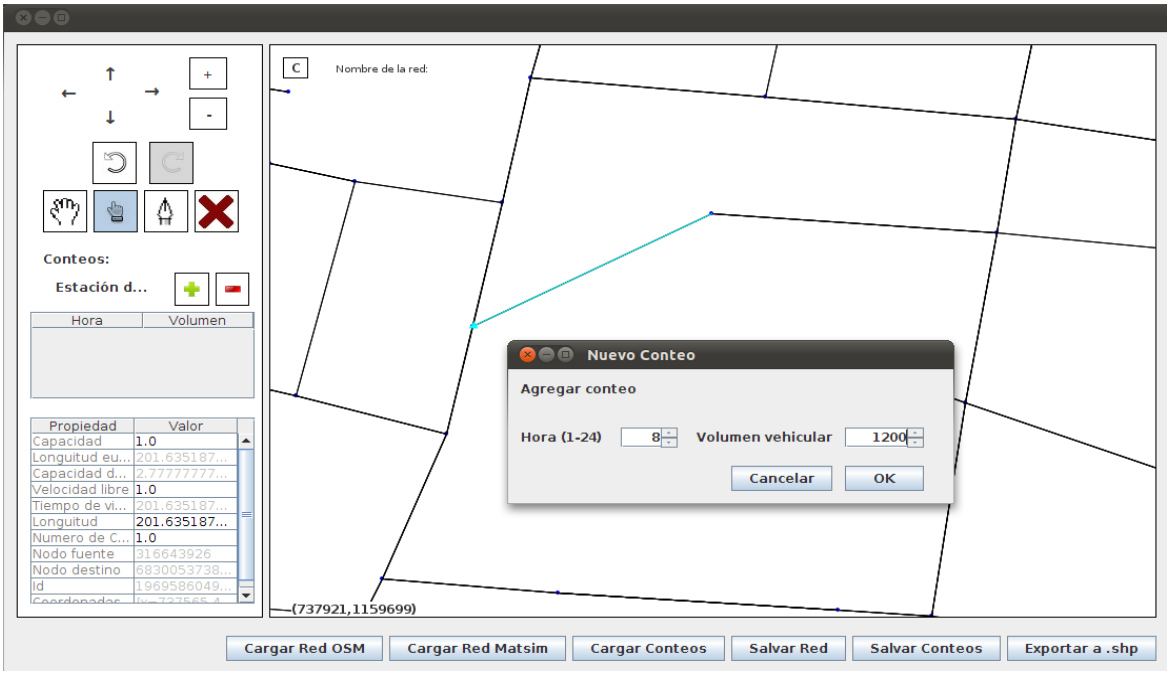


Figura 7.8: Creación de un nuevo conteo

7.3.5. Dehacer y rehacer cambios hechos sobre la red

Se procederá a eliminar el enlace creado utilizando la funcionalidad *deshacer*. En la figura 7.9 se puede apreciar el resultado de la operación. De igual forma, se puede observar el ícono de la función *rehacer* activo.

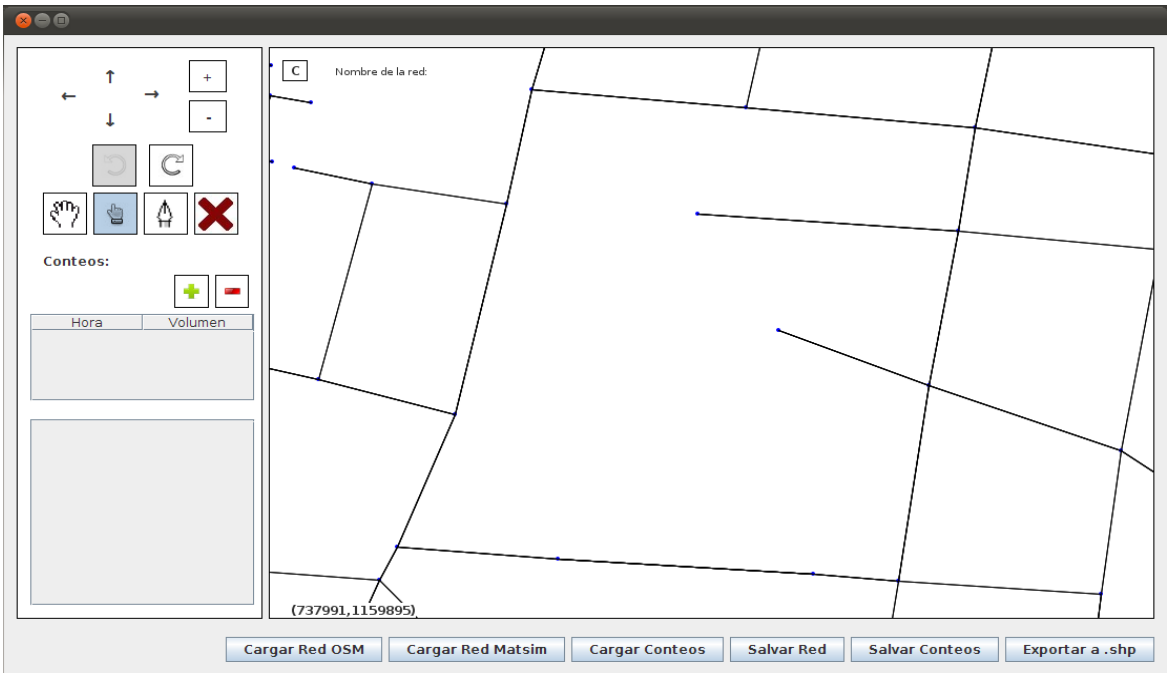


Figura 7.9: Deshacer la acción de crear un nuevo enlace

Capítulo 8

Modelo de tráfico de Los Cortijos de Lourdes y Los Ruices

Habiendo sido escogida la herramienta a utilizar para realizar el proceso de simulación (MATSim), se hizo necesario seleccionar un escenario particular en Caracas sobre el cual se puedan realizar simulaciones. El principal problema que se debe enfrentar para la escogencia de la zona es el acceso a la información de tráfico. Como se vio en la sección 4.4, el proceso del análisis de la demanda de tráfico requiere la realización de un estudio intensivo sobre la población que vive o labora en una zona determinada y sus hábitos de transporte, bien mediante encuestas y/o información de censo.

A través de la Asociación de Industriales de Los Cortijos y Los Ruices (ASICOR), se tuvo acceso a una investigación de tráfico realizada en la zona de Los Cortijos y Los Ruices por la empresa URVISA, bajo la solicitud de ASICOR, efectuada en el año 2006 y con una actualización realizada en el año 2008. Esta investigación realizada por URVISA permitió el acceso a varios datos significativos y fundamentales para efectos del presente trabajo de grado:

- Información detallada del trazado de calles y parcelas de la zona.
- Información de la población que trabaja y vive en la zona, específicamente de sus hábitos de transporte.

- Información sobre los viajes producidos y atraídos por cada parcela de la zona de Los Cortijos y Los Ruices.
- Matriz origen destino de la zona estudiada.
- Volúmenes de tráfico asignados a cada enlace de la red por el modelo de asignación usadoen el estudio de URVISA.
- Conteos mecánicos y clasificados de las principales calles de la zona estudiada.

La obtención de estos datos permitió aligerar considerablemente la carga de trabajo de campo del presente trabajo de tesis, logrando que éste se enfocara fundamentalmente en los objetivos expuestos en el planteamiento del problema.

8.1. Análisis de demanda de transporte en la zona de Los Cortijos de Lourdes y Los Ruices

Esta sección repasará los pasos seguidos para generar los datos de demanda de transporte suplidos a la simulación sin entran en muchos detalles sobre la implementación de los algoritmos que se usaron para ello.

Como se reseña en la sección 4.4, la generación de los datos de demanda para alimentar la entrada de un simulador, requiere que se siga el modelo de 4 pasos:

1. Generación de tráfico
2. Distribución de tráfico
3. Elección de modo de transporte
4. Asignación de tráfico

URVISA, facilitó la elaboración de los dos primeros pasos, en tanto que MATSim proveerá las herramientas para realizar la cuarta tarea. La tercera tarea se omite puesto que el estudio aplicado a la zona de interés son relativos a tráfico vehicular y no peatonal.

8.1.1. Estudio de vialidad realizado por URVISA

Como se indicó al comienzo del capítulo, ASICOR y URVISA facilitaron un estudio realizado por éste último en la zona industrial de Los Cortijos de Lourdes y Los Ruices. Esta zona casi entéramente industrial y comercial, posee algunas características propias:

- Es una zona casi nétamente atractora de viajes en las horas de la mañana. Por otro lado, se convierte en una zona productora de viajes en horas vespertinas.
- El grueso de las personas que laboran en la zona llegan a sus puestos de trabajo en Metro o en vehículo particular, representando el transporte público terrestre menos del 10 % del volumen vehicular diario.
- La zona tiene acceso desde 2 vías troncales: la autopista Francisco Fajardo y la Avenida Francisco de Miranda. De forma indirecta, la zona recibe tráfico de la Avenida Rómulo Gallegos.

8.1.1.1. Generación de la demanda de tráfico

Para obtener los datos poblacionales de la zona de Los Cortijos de Lourdes y Los Ruices, ASICOR elaboró encuestas que se repartieron a las empresas localizadas en las parcelas de la zona. A falta de una encuesta total o inventario de viajes completo por cada parcela, se elaboró la tabla de origen destino usando datos demográficos suministrados por ASICOR. De esta forma se obtuvieron índices de viajes por metro

cuadrado de construcción, lo cual se logró estimando a partir de el número de pisos reportados o inventariados. Finalmente se expandieron, por aproximaciones sucesivas, estas estimaciones a todas las parcelas[80].

Una vez obtenida la información de la cantidad de viajes estimados por parcela, se dividió la zona de Los Cortijos y Los Ruices en 59 sectores, tanto internos como externos, siendo los sectores externos principalmente productore y los internos consumidores. No obstante, es necesario que los sectores externos también sean consumidoras, de tal forma que se pueda simular los desplazamientos entre sectores externos, i.e., vehículos que transitan a través de la zona pero no realizan ninguna actividad dentro de ella.

A cada sector se le asignaron valores de origen y destino, generando así la tabla descrita en el apéndice I.

8.1.1.2. Distribución de la demanda de tráfico

Habiendo dividido la zona Los Ruices y Los Cortijos en los 59 sectores y habiendo asignado valores de origen y destino a cada sector, lo siguiente que se hizo fue generar una matriz origen destino que refleje los viajes interzonales realizados. Para ello, se aplicó un modelo gravitatorio el cual está definido como:

$$T_{ij} = \frac{P_i A_j \frac{1}{C_{ij}^x}}{\sum_{i=1}^{N} A_j \frac{1}{C_{ij}^x}}$$

donde:

T_{ij} : viajes entre las zonas (i, j)

P_i : viajes producidos por la zona i

A_j : viajes atraídos por la zona j

C_{ij} : costo de desplazarse de la zona i a la zona j

El valor calibrado de x usado está en el rango $1{,}5 \leq x \leq 2$.

En el apéndice J se encuentra la matriz origen destino generada por este modelo mediante el software *Motors* usado por URVISA.

8.1.1.3. Otros datos provistos por URVISA

URVISA también realizó el paso de asignación de tráfico con los datos obtenidos a partir de la generación y distribución de la demanda de tránsito, obteniendo valores de volumen de tráfico para cada uno de los enlaces en la horas pico. Este proceso fue iterativo, puesto que fue necesario ajustar los datos obtenidos en los pasos de generación y distribución de tal forma que los valores de volúmenes asignados en cada enlace concordaran con los volúmenes reales obtenidos por las estaciones de conteo [80]. En la figura 8.1 es posible encontrar la ubicación de las estaciónes de conteo electromecánicos colocadas por URVISA en algunas avenidas y calles clave de la zona de Los Cortijos y Los Ruices.

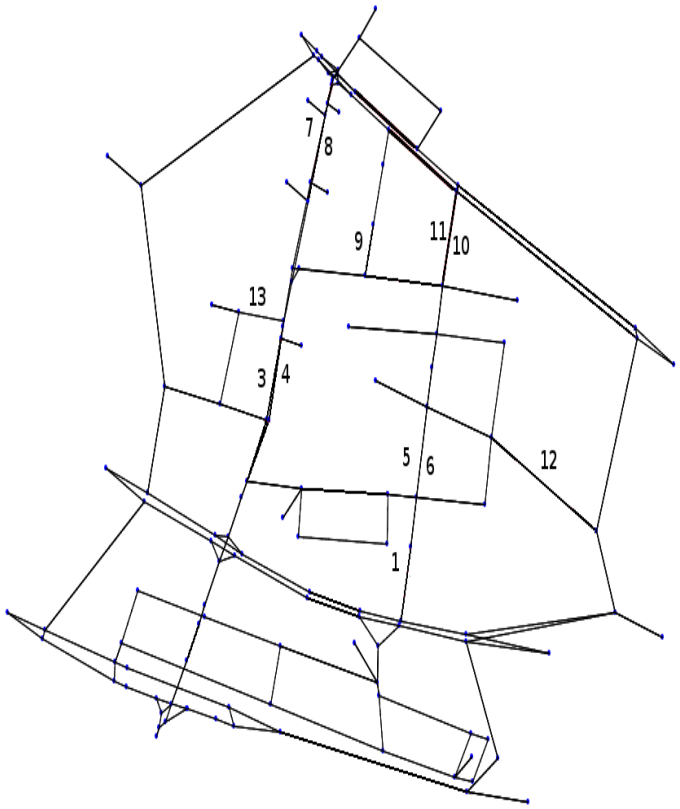


Figura 8.1: Ubicación de las estaciones de conteo

El modelo de asignación usado por URVISA fue el *modelo de asignación incremental* (véase 4.4.3), resultando los valores de flujo por enlace en las horas de la mañana los descritos en el apéndice K. Estos valores servirán para validar la simulación hecha por MATSim.

8.2. Convertir una matriz origen destino en planes de MATSim

Puesto que MATSim necesita que la demanda sea reflejada en planes (véase 5.1), es necesario un programa que traduzca la información de la matriz origen destino en planes. Para lograr este cometido, se diseñó un modelo basado en sectores y enlaces

asociados a éstos. La figura 8.2 muestra el diagrama de las clases involucradas de convertir las matrices origen destino en planes.

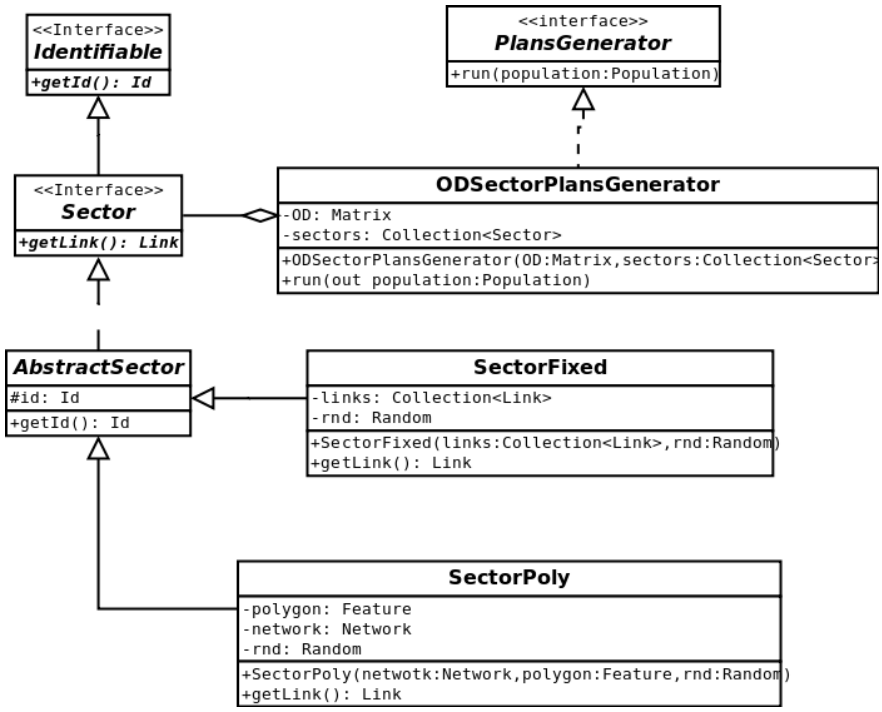


Figura 8.2: Diagrama de clases del generador de planes a partir de matrices origen destino

La interfaz *Sector* únicamente provee el método *getLink()*, el cual devuelve un enlace aleatorio perteneciente a éste.

El algoritmo fundamental para producir el archivo de planes matutinos a partir de la matriz origen destino de horas AM y la información de los sectores, es el siguiente:

PARA cada par de zonas (i, j) de la matriz origen destino M **HACER**
  Tomar un enlace aleatorio L1 de la zona i
  Tomar un enlace aleatorio L2 de la zona j
  Crear M(i, j) personas con un unico plan cuya actividad
  "home" sea en el enlace L1 y la actividad "work" en el
      enlace L2.
FIN PARA

Un inconveniente derivado de la obtención de los datos de viajes a partir de la matriz origen destino es la asignación de horas de comienzo a cada actividad porque no se poseen datos detallados, por lo tanto el generador de demanda establecerá la hora de llegada a las actividades a partir de una distribución normal con media μ y desviación σ.

Si bien está planteada en el diseño la clase SectorPoly, esta no está completamente implementada. Para el momento de la redacción de este documento, la clase SectorFixed es la única de las dos clases que implementan la interfaz Sector que está completa.

A continuación se explicará el método usado por cada uno de los tipos de sector: *SectorPoly* y *SectorFixed*.

8.2.1. La clase *SectorPoly*

La clase *SectorPoly* genera los planes a partir de la siguiente entrada:

- Un archivo de texto plano representando la matriz origen destino.
- Un archivo en formato SHP que contenga los polígonos representativos de las sub-zonas inherentes a la zona de estudio. Cada polígono debe tener establecido un atributo entero *id* que representa su identificador.
- Una red en formato MATSim.

Existen dos posibles métodos para la selección del enlace aleatorio perteneciente a una zona:

1. Generar un punto aleatorio dentro del polígono relativo a la zona y buscar el enlace más cercano a ese punto.
2. Pre-procesar la red y tener almacenado con anterioridad por cada polígono los enlaces que lo tocan, seleccionando cualquiera de ellos de forma aleatoria cuando sea necesario obtener un enlace.

8.2.2. La clase *SectorFixed*

La clase *SectorFixed* genera los planes a partir de la siguiente entrada:

- Un archivo de texto plano representando la matriz origen destino.
- Un archivo de texto plano con información que relaciona los enlaces de la red con zonas.
- Una red en formato MATSim.

El formato del archivo de texto que contiene las relaciones entre los enlaces y las zonas posee el siguiente formato:

```
<zona 1> <enlace 1> <enlace 2> ... <enlace n>
<zona 2> <enlace 1> <enlace 2> ... <enlace n>
                  ...
<zona m> <enlace 1> <enlace 2> ... <enlace n>
```

donde los enlaces y las zonas estarán representadas por los identificadores (id) de cada uno de ellos, i.e., las zonas enumeradas por URVISA y los enlaces de la red, en formato MATSim, que se esté usando.

Capítulo 9

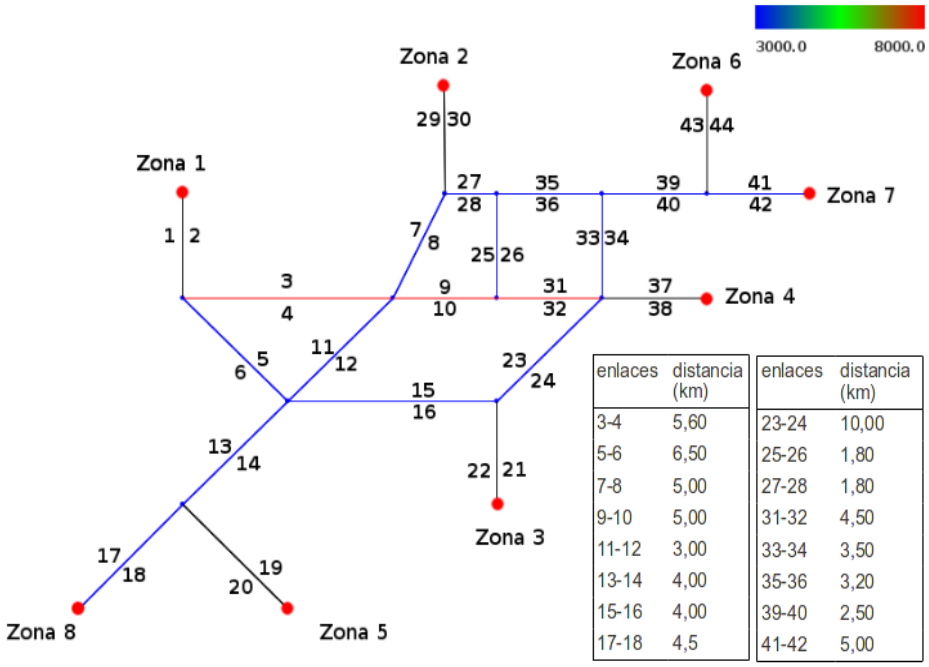
Experimentos y resultados

9.1. Pruebas preliminares: un ejemplo sencillo

La primera prueba realizada con MATSim fue sobre una red pequeña, la cual consta de 8 zonas productoras/generadoras con sus respectivos centroides. La figura 9.1 representa la red gráficamente, donde los puntos rojos representan los centroides de cada una de las 8 zonas.

La red está conectada por una autopista (en rojo) y varias avenidas (en azul), con valores de capacidad de 8000 y 3000 vehículos por hora respectivamente. Los centroides están conectados a la red a través de enlaces artificiales con capacidad infinita (en negro), i.e., un valor arbitráriamente alto.

Por otro lado, se construyó una matriz origen destino artificial. La tabla 9.1 representa la distribución de viajes usada para este ejemplo.



enlaces	distancia (km)
3-4	5,60
5-6	6,50
7-8	5,00
9-10	5,00
11-12	3,00
13-14	4,00
15-16	4,00
17-18	4,5

enlaces	distancia (km)
23-24	10,00
25-26	1,80
27-28	1,80
31-32	4,50
33-34	3,50
35-36	3,20
39-40	2,50
41-42	5,00

Figura 9.1: Red de ejemplo

Zonas	1	2	3	4	5	6	7	8
1	0	350	1000	120	0	0	50	70
2	150	0	600	100	0	0	80	60
3	250	750	0	300	0	0	60	60
4	320	150	2500	0	0	0	100	60
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	200	160	400	350	0	0	0	1200
8	350	150	600	180	0	0	1500	0

Tabla 9.1: Matriz origen destino de ejemplo

Las zonas 1, 2, 3 y 4 representan zonas internas con desarrollo urbanístico. Las

zonas 5 y 6 no producen ni consumen viajes por cuanto son zonas para desarrollo a futuro. Finalmente, las zonas 7 y 8 representan zonas externas.

Para generar los planes se usó el método de generación por sectores fijos (véase 8.2.2), asociando los enlaces correspodientes con sus centroides de zonas. Los parámetros de la normal que establece la hora de salida de la actividad *home* en cada plan fueron establecidos en $\mu = 8$ y $\sigma = 0{,}5$.

Una vez establecido el escenario de simulación, se procedió a simular usando los siguientes parámetros en el archivo de configuración de MATSim:

Número de iteraciones	50
Probabilidad de mantener el plan de mejor puntuación	0.5
Probabilidad de cambiar la ruta	0.1
Probabilidad de cambiar la hora de salida de la actividad *home*	0.4

Una vez realizada la simulación, se obtuvieron las siguientes estadísticas de desempeño:

Tiempo de ejecución total 3:37

Tiempo de ejecución de usuario 3:30

Tiempo de ejecución del sistema 00:04

La figura 9.2 muestra el resultado de la puntación de los planes, donde se puede apreciar que no se obtuvo mucha mejora por la ejecución de las iteraciones. Por otro lado, esta simulación sirvió para constatar la asignación de rutas iniciales, la cual sigue el modelo de asignación "todo o nada" (véase 4.4.3). La figura 9.3 muestra el histograma de eventos generado en la última iteración. Finalmente, los volúmenes asignados a cada enlace se encuentran plasmados en la tabla 9.2

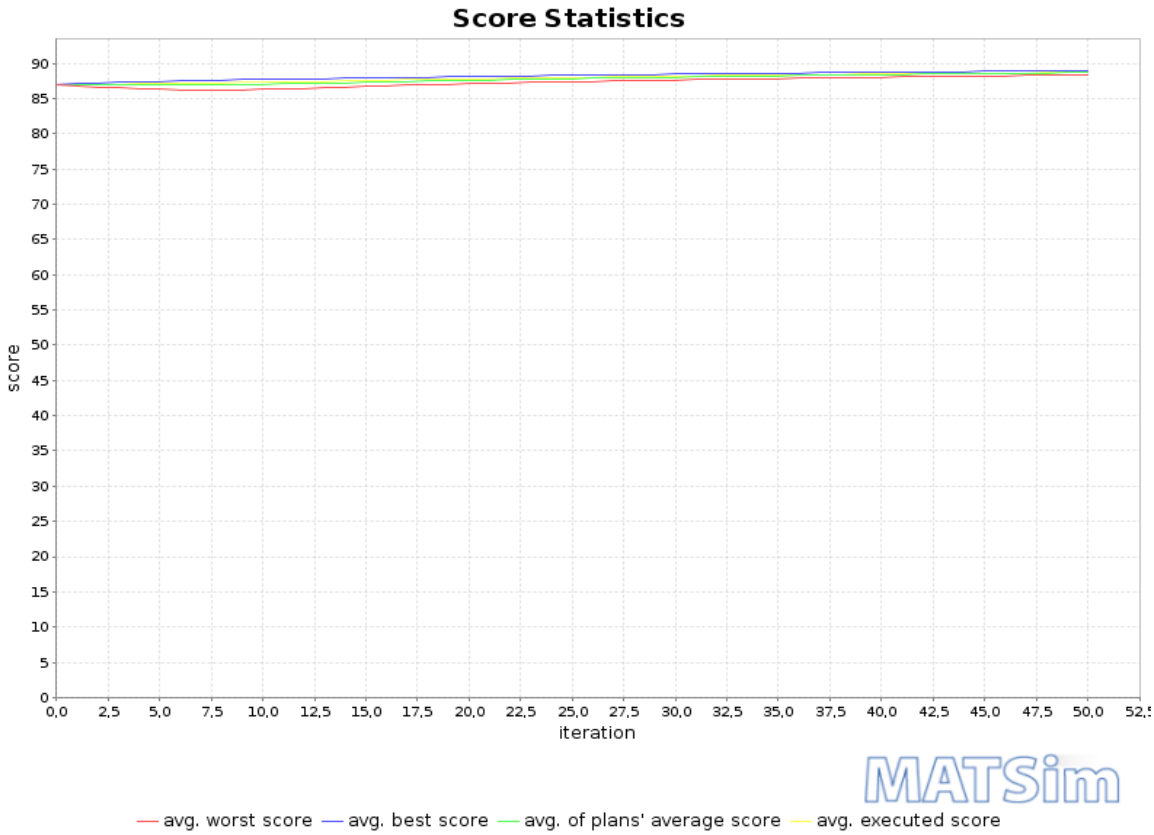


Figura 9.2: Puntuación de la ejecución de los planes para la simulación de ejemplo

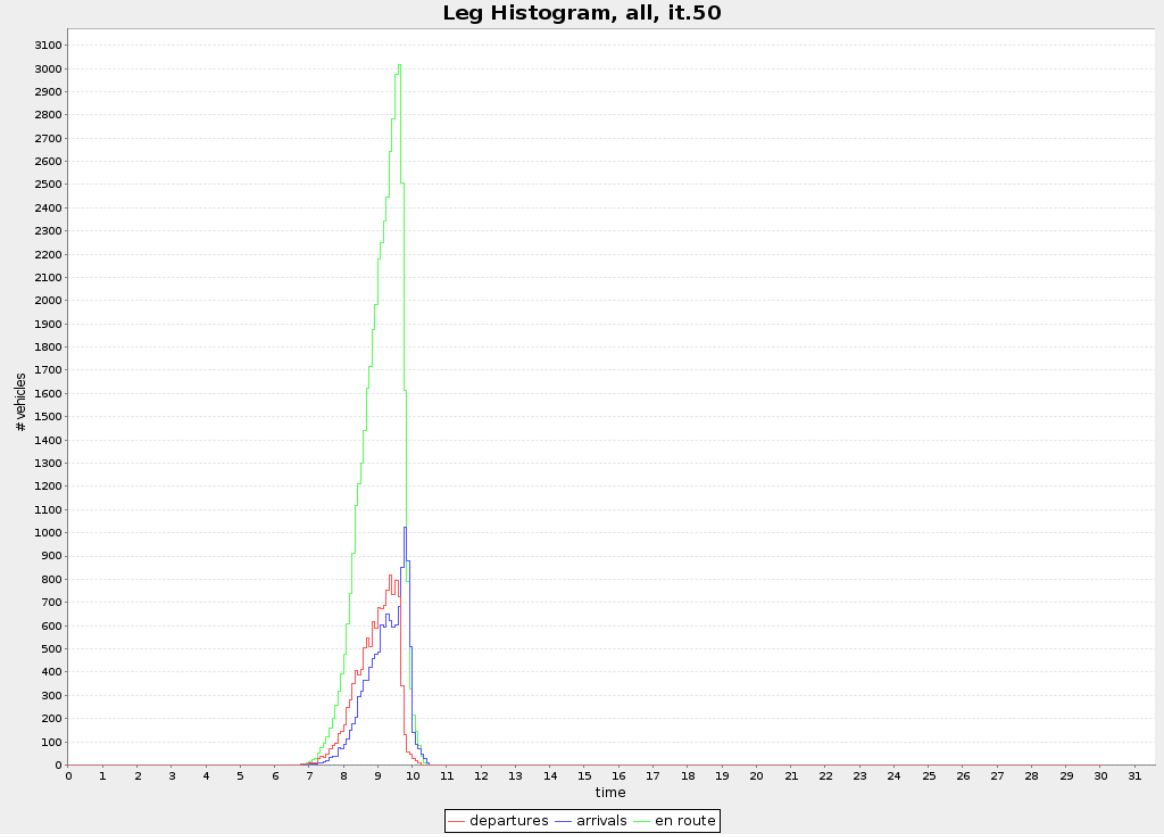


Figura 9.3: Histograma de eventos para la simulación de ejemplo

Enlace	Volumen
1	1590
2	1590
3	864
4	857
5	726
6	413
7	1250
8	810
9	1850
10	1780
11	2264
12	2767
13	1450
14	2780
15	2200
16	1060
17	1450
18	2780
21	5100
22	1420
23	360
24	2900
25	1700
26	1500
27	180
28	310
29	1560
30	990
31	400
32	530
33	160
34	750
35	1630
36	1560
37	1050
38	3130
39	1790
40	2310
41	1790
42	2310

Tabla 9.2: Volúmenes obtenidos por enlace

9.2. Preparación de la simulación de tŕafico de Los Cortijos de Lourdes y Los Ruices

9.2.1. Red víal

Antes de comenzar a simular el tráfico en la zona de Los Cortijos de Lourdes y Los Ruices, se debió modificar la red de tal forma que contuviera los centroides de algunos sectores, especialmente de los sectores externos, además de incluir las viás alternas de acceso. La figura 9.4 muestra la red modificad de la zona de estudio, con las capacidades establecidas de acuerdo a la escala de color. Los enlaces de color negro indican enlaces artificiales con capacidad "infinita" que conectan los centroides con la red. Los enlaces que representan las vías alternas o "caminos verdes", fueron dibujados sin relación estricta con la topología real de la red, sin embargo, su capacidad y longitud fue establecida en relación con la capacidad y la longitud real de éstos. Por otro lado, los centroides de los sectores internos fueron representados con una combinación de enlaces artificiales y enlaces pertenecientes a la red red real. Puesto que en MATSim los órigenes y destinos de las actividades son enlaces, se requirío en algunos casos que se crearan los enlaces artificiales, en especial para aquellos centroides que estaban establecidos en esquinas o intersecciones, o los que se ubicaban en calles/avenidas largas, para lo cual se crearon dos enlaces artificiales conectando la calle, uno de ida y otro de vuelta.

La tabla 9.3 muestra los valores de capacidad y velocidad de libre establecidos en los enlaces de la red.

Los valores de capacidad y velocidad libre fueron establecidos de acuerdo con información provista por URVISA y por el conversor de redes OSM de MATSim. Por otro lado, los valores establecidos en los enlaces que representan las vías alternas fueron

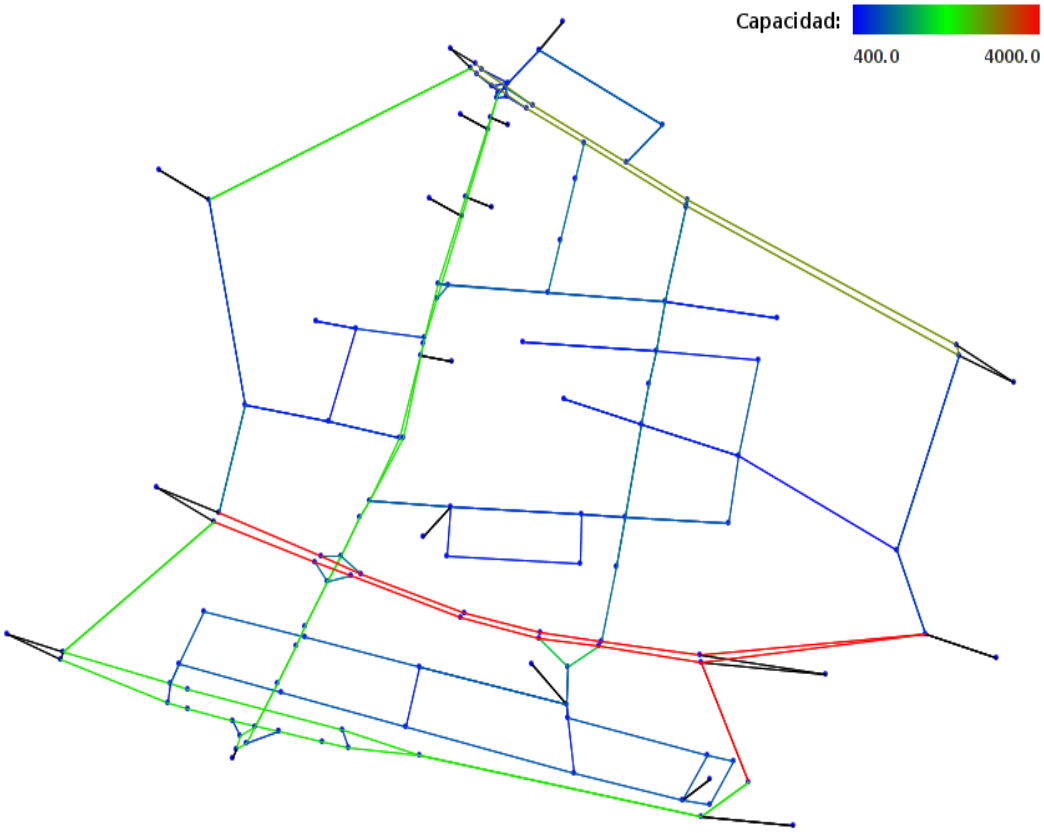


Figura 9.4: Red modificada de Los Cortijos de Lourdes y Los Ruices

Tipo de vía	Capacidad (veh./hora)	Velocidad Libre (kph)	Ejemplo
Calle interna	600-800	30	2da. Trans. de Los Cortijos
Calle principal o Avenida	1000-1200	30-60	Avenida Alejandro Hernández
Avenida principal	2400-3000	60-80	Avenida Francisco de Miranda
Autopista	8000	120	Autopista Francisco Fajardo
Enlace artificial	6000000	360	N/A

Tabla 9.3: Valores establecidos de capacidad y velocidad libre de los enlaces

establecidos a de forma *ad-hoc*, mediante simulaciones sucesivas donde los resultados se contrastaban con los valores de flujo por enlace obtenidos por URVISA en su estudio.

9.2.2. Planes

URVISA suministró dos matrices origen destino de actividades laborales, una de las horas AM y otra de las horas PM, de las cuales fue escogida la primera para convertirla a planes y simular. La razón de peso para esta escogencia reside en el hecho que las horas de entrada al trabajo suelen ser menos variantes que la de salida.

Se generaron 3 conjuntos de planes, todos ellos mediante el método propuesto en la sección 8.2.2, con los siguientes parámetros para la finalización de la actividad *home*:

Plan 1 : $\mu = 7$, $\sigma = 0{,}25$

Plan 2 : $\mu = 7$, $\sigma = 0{,}50$

Plan 3 : $\mu = 7$, $\sigma = 0{,}1$

El valor tomado como media se fijó a partir de los experimentos previos que se realizaron con el fin de calibrar la red, siendo el valor de 7 AM el que mejor ajustaba los conteos de la simulación con los conteos reales.

Los planes ejecutados se denominaron *plans7_25.xml* , *plans7_50.xml* y *plans7_10.xml*, respectivamente.

9.2.3. Preparación del ambiente de simulación: configuración, estructura de directorios y *scripts*

9.2.3.1. Configuración

Se generaron 17 escenarios de simulación, cada uno con parámetros distintos, con el fin de probar el desempeño de cada uno de ellos computacionalmente y, lo

más importante, en la veracidad de los datos resultantes. La tabla 9.4 muestra los parámetros escogidos para cada uno de los escenarios de simulación. Como parámetro común a todos ellos, se estableció el número de iteraciones en 80 y la memoria de cada agente en 10 . En la tabla también es posible apreciar el valor de probabilidad asignado a cada estrategia. La estrategia *BestScore* está presente en todos los escenarios, donde:

$$EstrategiaBestScore = 1 - (EstrategiaReRoute + EstrategiaTimeAllocatorMutator)$$

Archivo de configuración	Archivo de Planes	Estrategia *ReRoute*	Estrategia *TimeAllocatorMutator*
config1.xml	plans7_50.xml	0,2	0
config2.xml	plans7_50.xml	0	0,2
config3.xml	plans7_25.xml	0,2	0
config4.xml	plans7_25.xml	0	0,2
config5.xml	plans7_50.xml	0,3	0,1
config6.xml	plans7_50.xml	0,1	0,3
config7.xml	plans7_50.xml	0,2	0,1
config8.xml	plans7_50.xml	0,1	0,2
config9.xml	plans7_50.xml	0,1	0,1
config10.xml	plans7_25.xml	0,3	0,1
config11.xml	plans7_25.xml	0,1	0,3
config12.xml	plans7_25.xml	0,2	0,1
config13.xml	plans7_25.xml	0,1	0,2
config14.xml	plans7_25.xml	0,1	0,1
config15.xml	plans7_10.xml	0,1	0,05
config16.xml	plans7_25.xml	0,1	0,05
config17.xml	plans7_50.xml	0,1	0,05

Tabla 9.4: Parámetros establecidos para cada uno de los escenarios de simulación

[0]El número máximo permitido de planes anteriores que puede recordar un agente

9.2.3.2. Estructura de directorios y *scripts*

Debido a la cantidad de escenarios a simular, era necesario una estructura de directorios que facilitara la clasificación de los resultados y su posterior análisis. Para tal fin, se creó la siguiente estructura de directorios:

res : contiene los recursos básicos para ejecutar la simulación, tal como la red, los planes y los conteos. También contiene archivos de configuración plantilla con el fin de facilitar la creación de nuevos archivos de configuración en el menor tiempo y con la menor cantidad de errores posible.

configs : dentro de este directorio es posible encontrar los archivos de configuración usados para cada simulación.

exec_times : contiene los tiempos de ejecución de cada simulación, medidos con el comando de UNIX *time*.

output_volumes : por cada simulación hecha, se recoge el volumen asignado a cada enlace de forma sumarizada. Este directorio contiene un archivo de texto plano por cada simulación hecha, con el siguiente formato por línea:

```
<id_enlace> <volumen_asignado>
```

output : los resultados generados por MATSim son colocados en este directorio.

Puesto que cada simulación puede demorar entre media hora a una hora en culminarse, se escribió un *scipt* que permitiera la ejecución automática de las simulaciones, respetando la estructura de directorios antes expuesta. El *script* permite la ejecución en serie de 1 o varios archivos de configuración.

9.3. Resultados de las simulaciones

9.3.1. Análisis y comparación de los resultados

Los resultados de la simulación a analizar serán los siguientes:

- Puntuación de los planes
- Histograma de eventos en la última iteración
- Conteos en las horas comprendidas entre 6 y 9 AM en la última iteración
- Volúmenes asignados a los enlaces en la última iteración
- Tiempo de ejecución

Los conteos serán contrastados con los conteos reales obtenidos mediante las estaciones de conteo colocadas por URVISA (véase figura 8.1) y los volúmenes obtenidos serán comparados con los volúmenes obtenidos por URVISA en su estudio realizado sobre al zona [80] en enlaces seleccionados. La figura 9.5 muestra los enlaces seleccionados para tal comparación con su respectiva numeración.

La gran cantidad de simulaciones realizadas hizo necesario que se escogieran sólo algunos escenarios a la hora de vizualisar los resultados obtenidos. Los escenarios escogidos, nominados por el archivo de configuración que los define, serán:

- Escenario 3: config3.xml
- Escenario 4: config4.xml
- Escenario 6: config4.xml

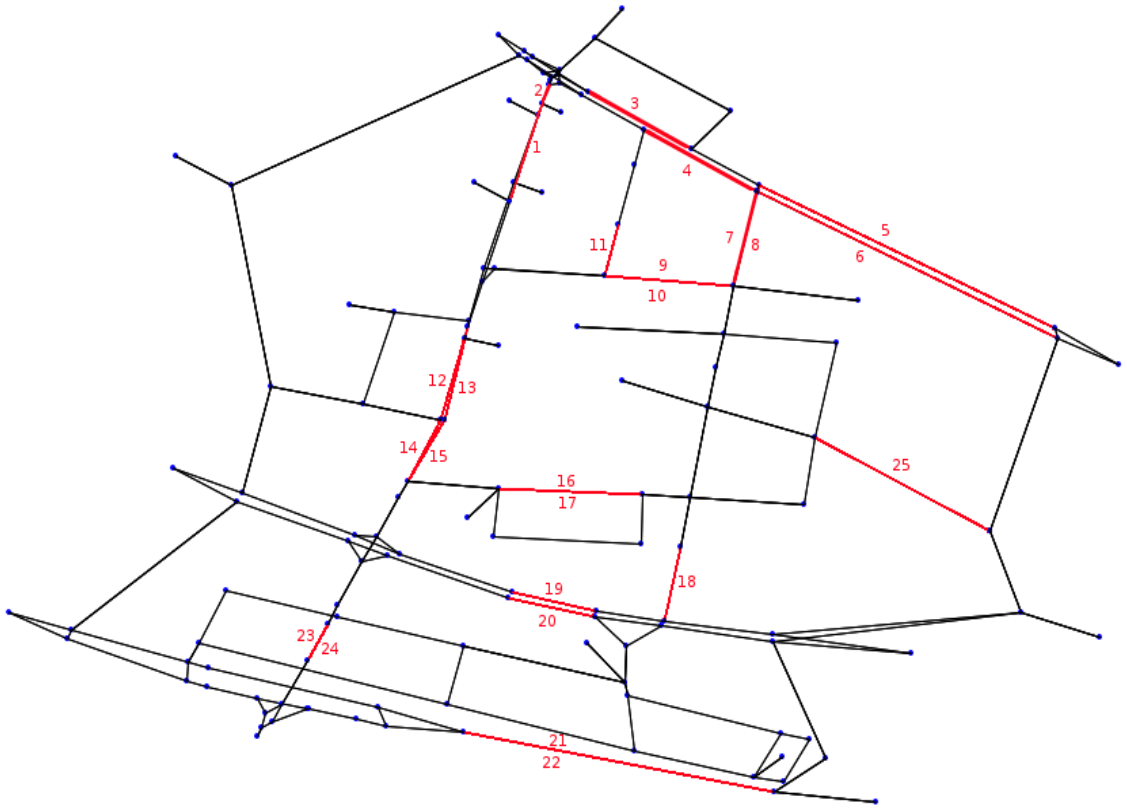


Figura 9.5: Enlaces seleccionados para comparar los flujos asignados por la simulación

9.3.2. Escenario 3

9.3.2.1. Puntuación de los planes

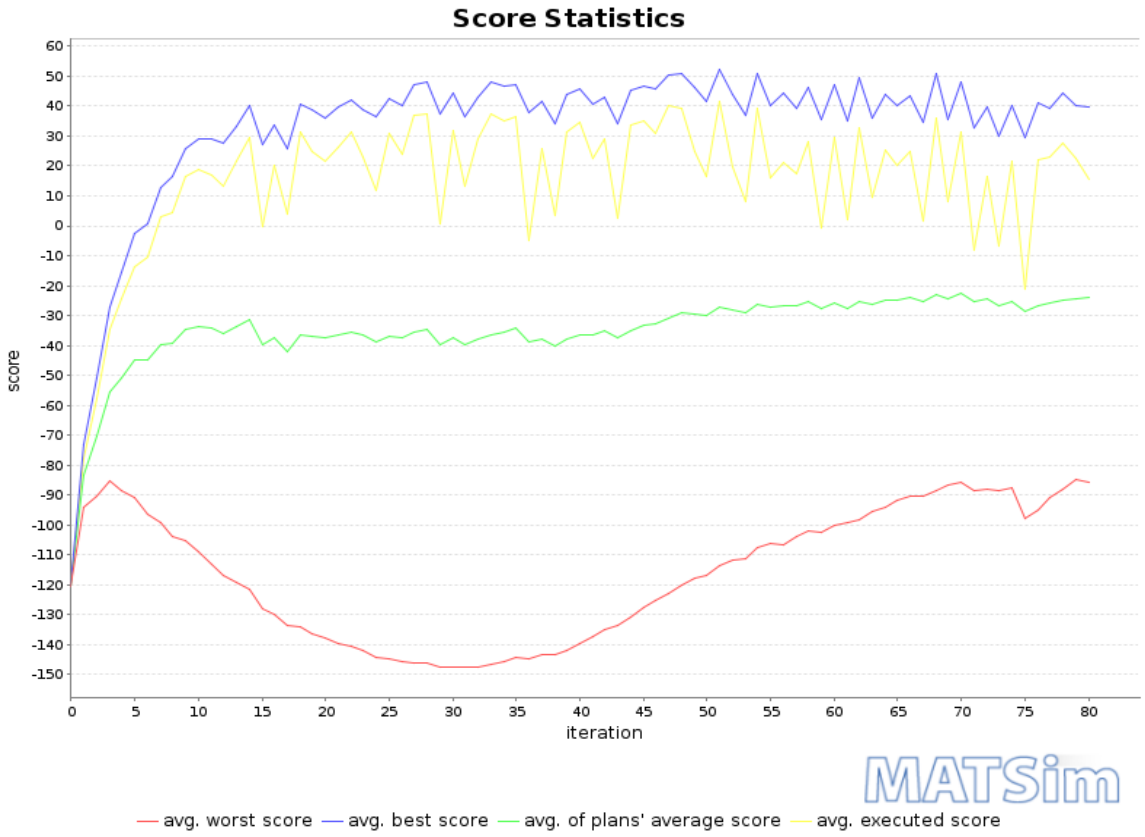


Figura 9.6: Puntación de los planes para el escenario 3

9.3.2.2. Histograma de eventos de la última iteración

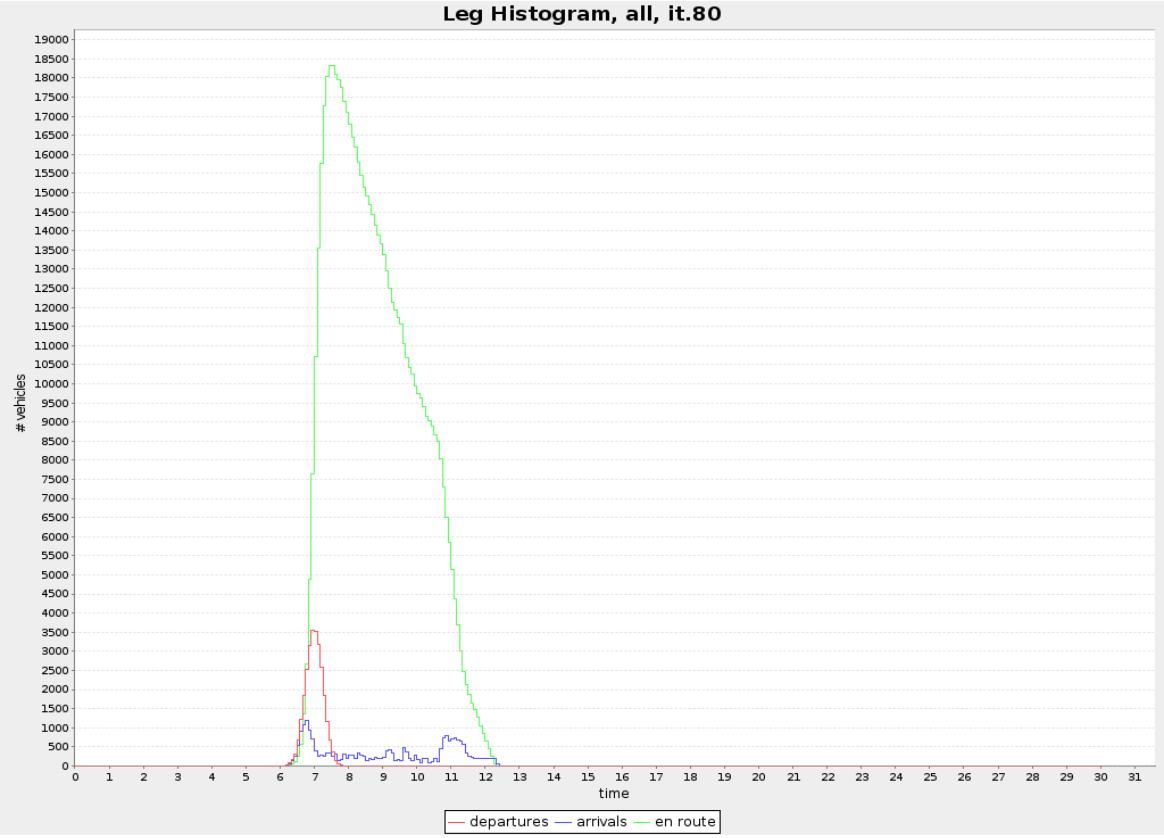


Figura 9.7: Histograma de eventos para el escenario 3

9.3.2.3. Conteos en las horas comprendidas entre 6 y 9 AM en la última iteración

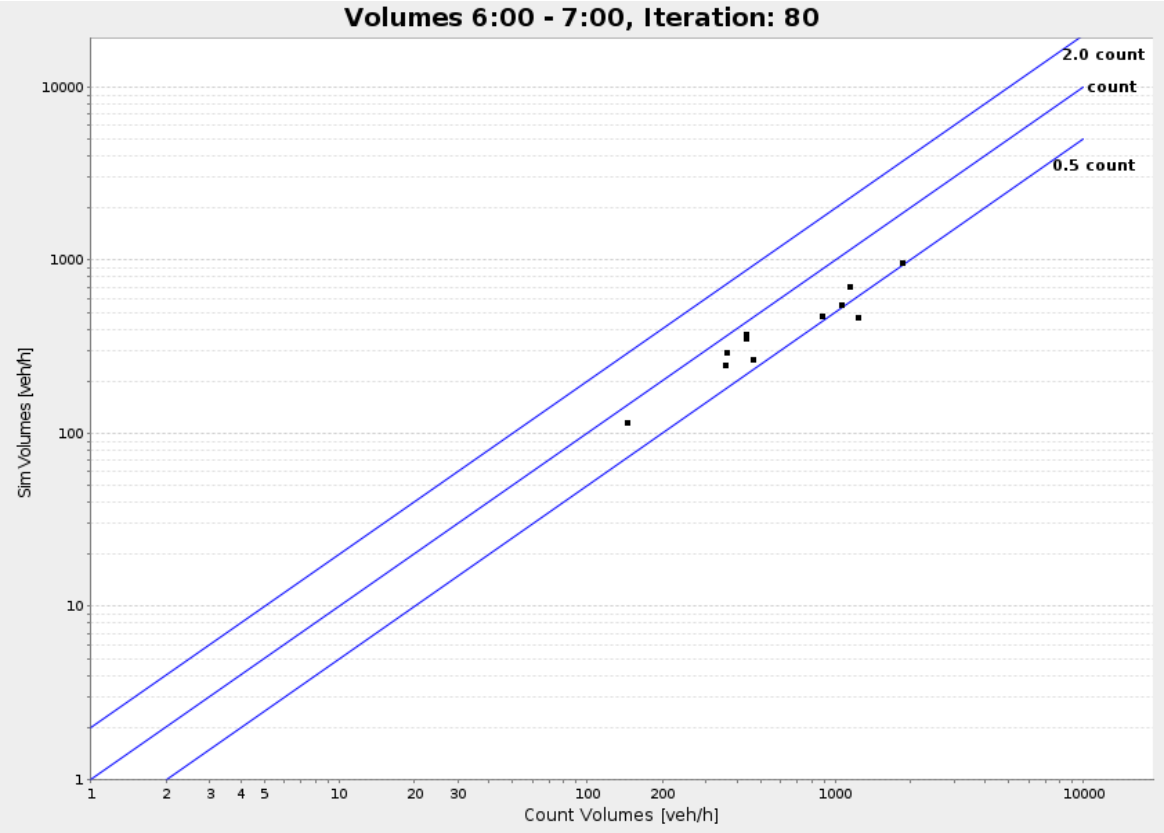


Figura 9.8: Conteos de 6-7 AM para el escenario 3

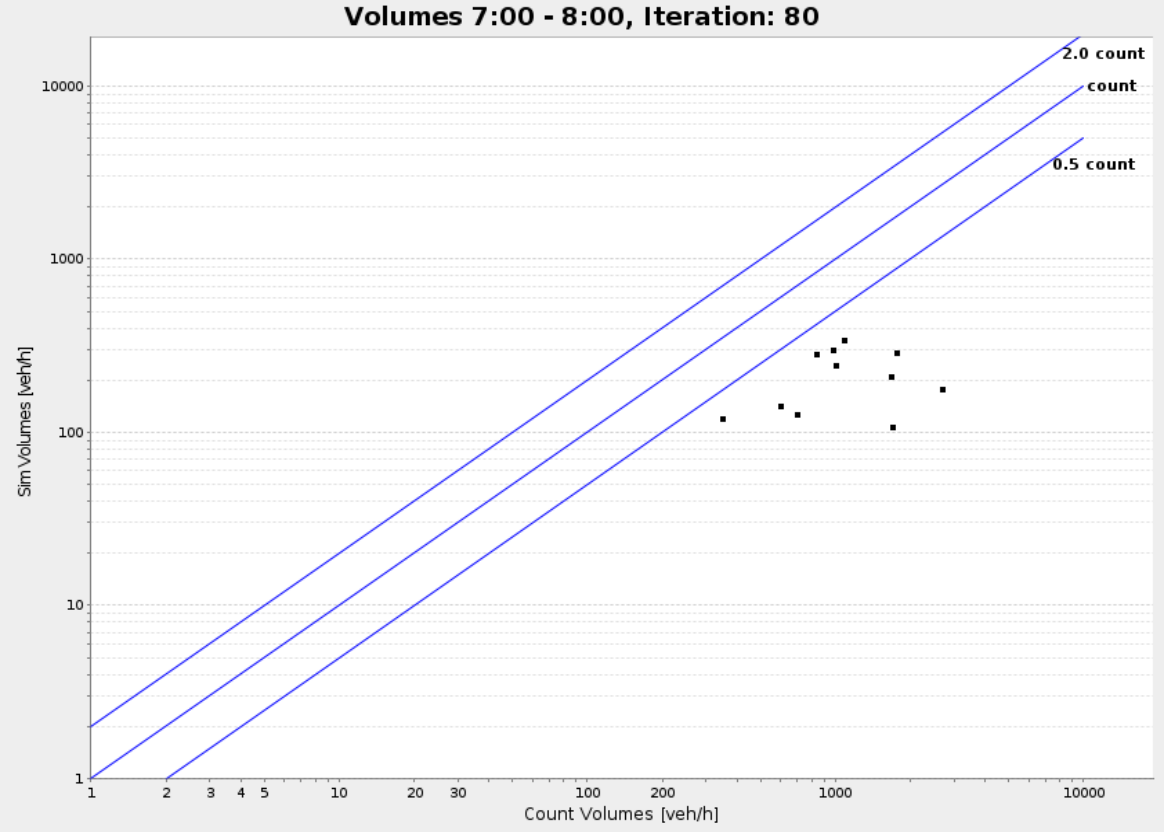


Figura 9.9: Conteos de 7-8 AM para el escenario 3

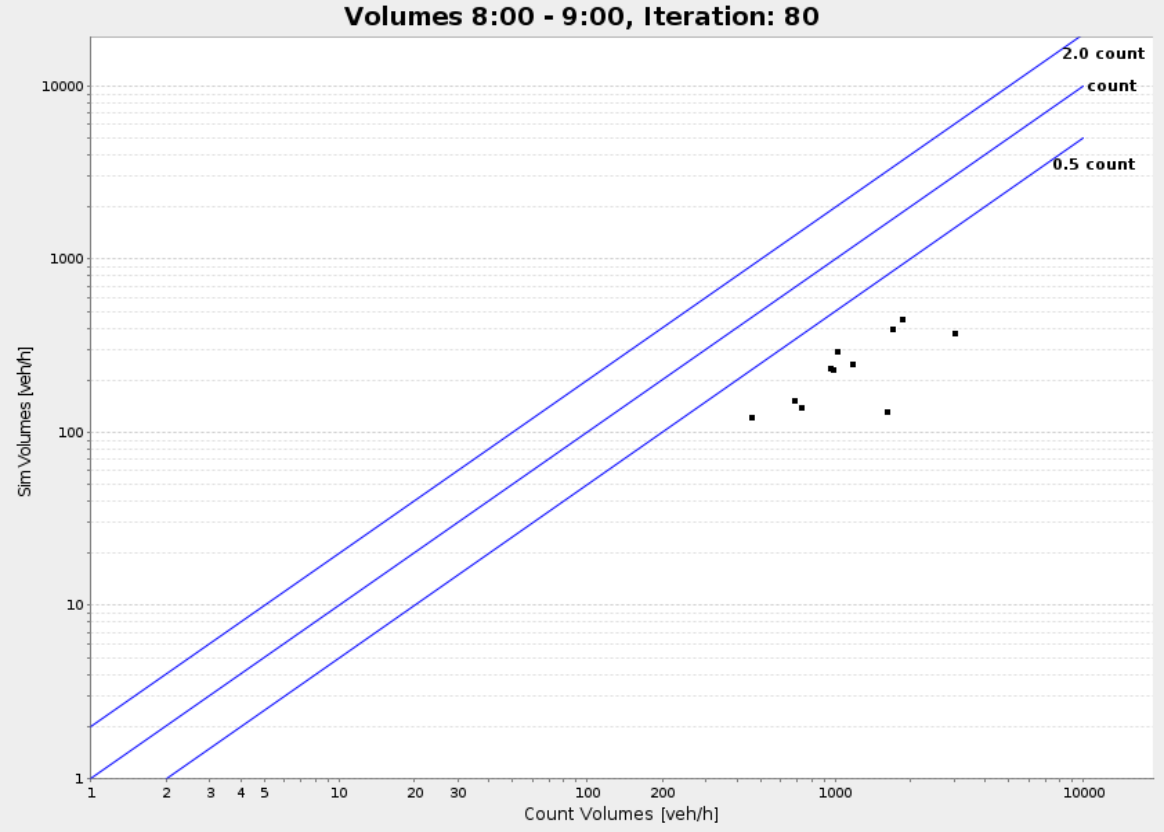


Figura 9.10: Conteos de 8-9 AM para el escenario 3

9.3.2.4. Volúmenes asignados a los enlaces en la última iteración

Enlace	Flujo obtenido	Flujo obtenido por URVISA	Diferencia Absoluta	Diferencia Relativa
1	2064	1027	1037	1,01
2	1969	1953	16	0,01
3	1685	2430	−745	−0,31
4	1908	1832	76	0,04
5	1565	2857	−1292	−0,45
6	1167	1649	−482	−0,29
7	1731	1247	484	0,39
8	1282	1285	−3	−0,00
9	405	612	−207	−0,34
10	763	1051	−288	−0,27
11	699	992	−293	−0,30
12	2304	1759	545	0,31
13	4376	1962	2414	1,23
14	2775	1504	1271	0,85
15	4376	2257	2119	0,94
16	1667	747	920	1,23
17	1719	1471	248	0,17
18	1740	1481	259	0,17
19	4791	6300	−1509	−0,24
20	4186	5736	−1550	−0,27
21	1718	1486	232	0,16
22	1075	309	766	2,48
23	1412	1457	−45	−0,03
24	2132	1990	142	0,07
25	713	719	−6	−0,01
			Prom. Dif. Absoluta	677,96
			Prom. Dif. Relativa	0,46

Tabla 9.5: Tabla comparativa de flujos para el escenario 3

9.3.2.5. Tiempo de ejecución

Los resutados obtenidos por el comando *time*, fueron los siguientes:

```
real 41m56.117s
user 40m55.845s
sys 0m14.629s
```

9.3.2.6. Análisis de los resultados obtenidos

Aún cuando los resultados de la puntuación de los planes son bastante modestos y la forma del histograma no se presenta realista, los conteos y los flujos asignados no están muy alejados de los valores obtenidos de los estudios previos realizados por URVISA, siendo de hecho, el escenario con el promedio de error relativo más bajo entre todos los escenarios de simulación. La aplicación de la estretegia pura de cambio de ruta (*ReRoute*) obliga a los agentes a salir todos casi a la misma hora, provocando un congestionamiento que se prolonga hasta el mediodía. Este escenario, donde la gente llega a trabajar tan tarde no es ajustado a la realidad, sin embargo, demuestra que la red actual no tiene las condiciones necesarias para soportar la carga vehicular en la hora pico si todos los vehículos estuvieran transitando por ella al mismo momento.

El tiempo promedio de viaje en este escenario fue de 02h:06m:38s y la distancia promedio recorrida por cada agente fue 1727.83 metros.

9.3.3. Escenario 4

9.3.3.1. Puntuación de los planes

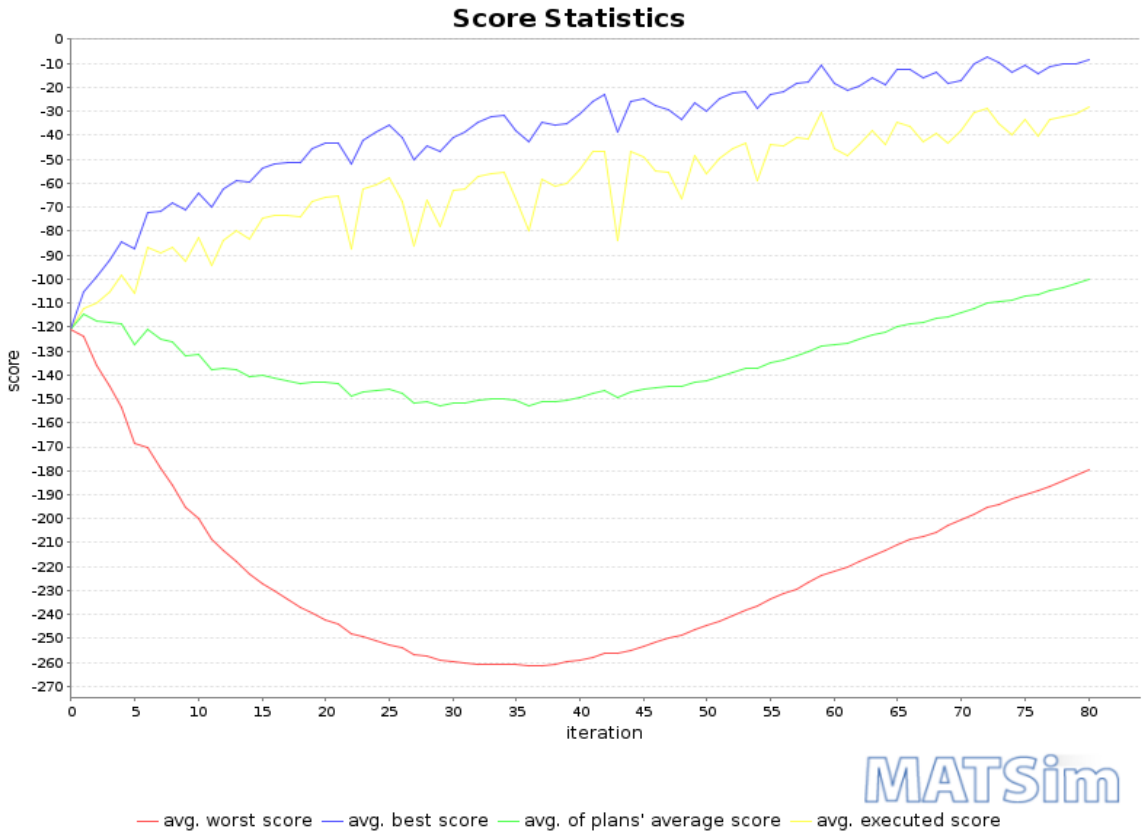


Figura 9.11: Puntación de los planes para el escenario 4

9.3.3.2. Histograma de eventos de la última iteración

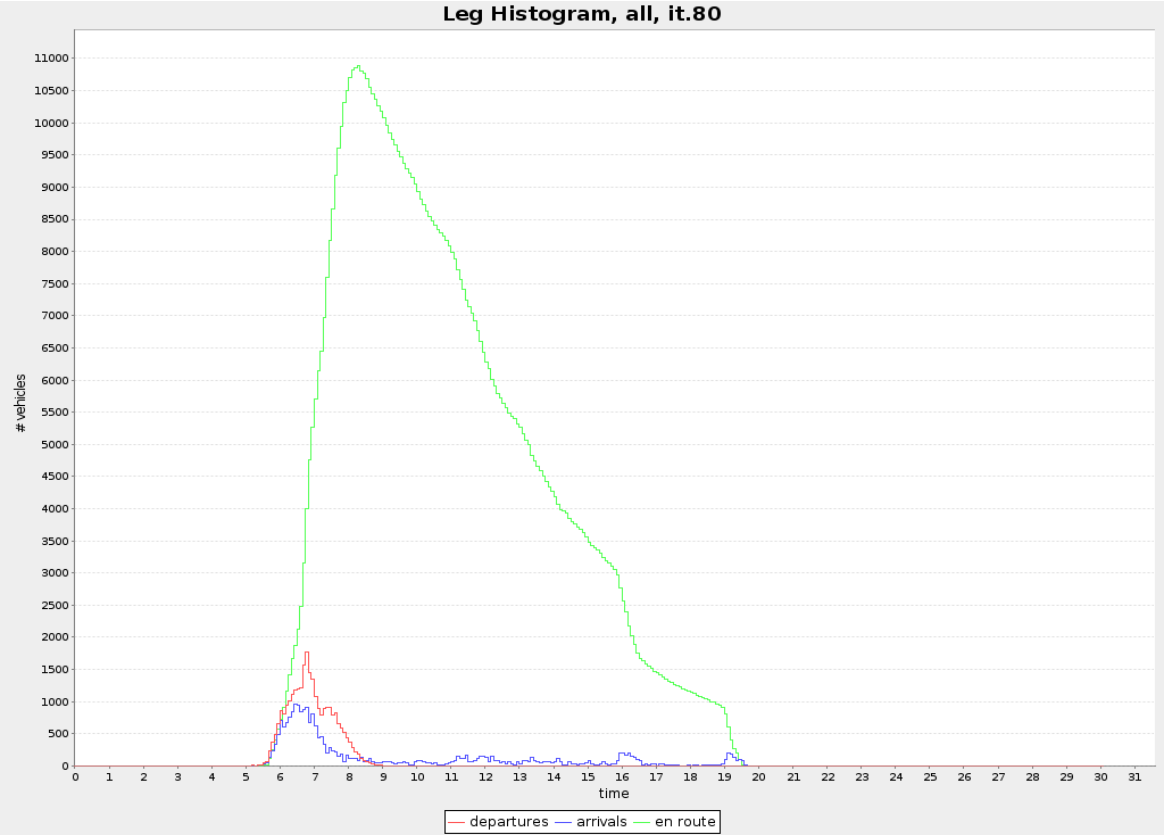


Figura 9.12: Histograma de eventos para el escenario 4

9.3.3.3. Conteos en las horas comprendidas entre 6 y 9 AM en la última iteración

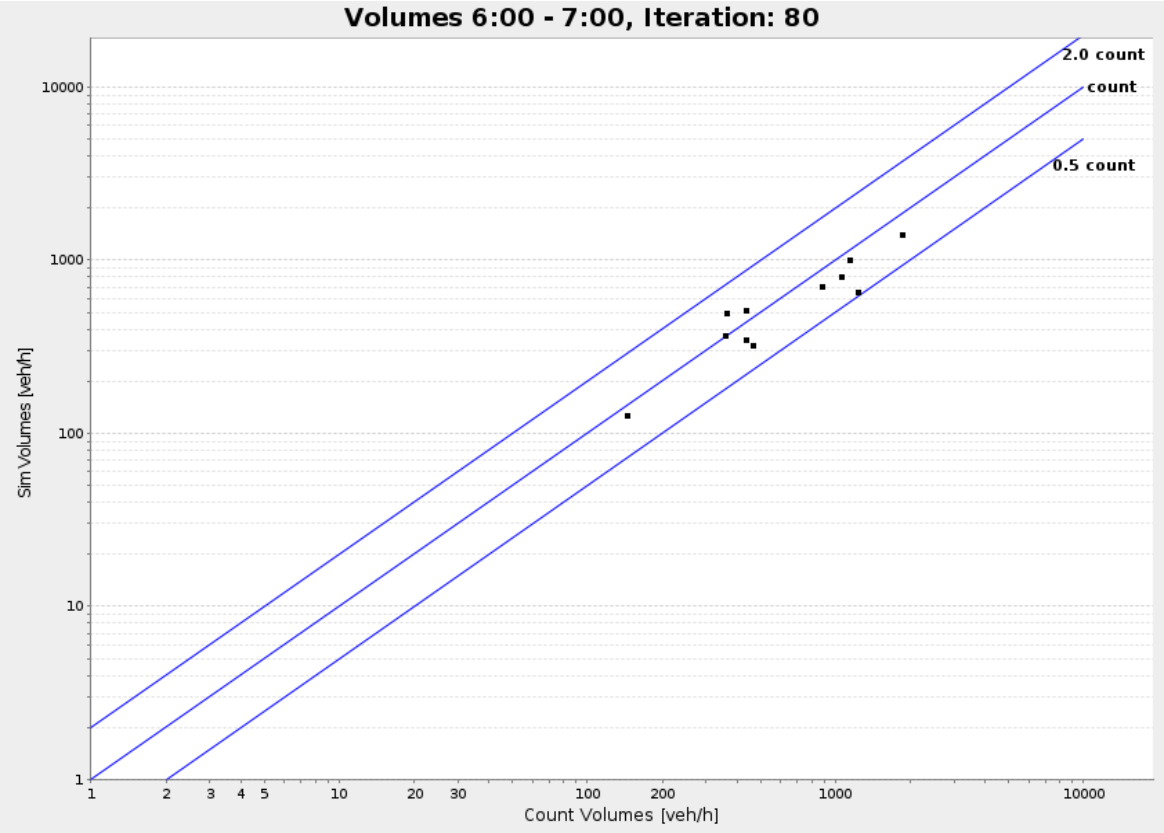


Figura 9.13: Conteos de 6-7 AM para el escenario 4

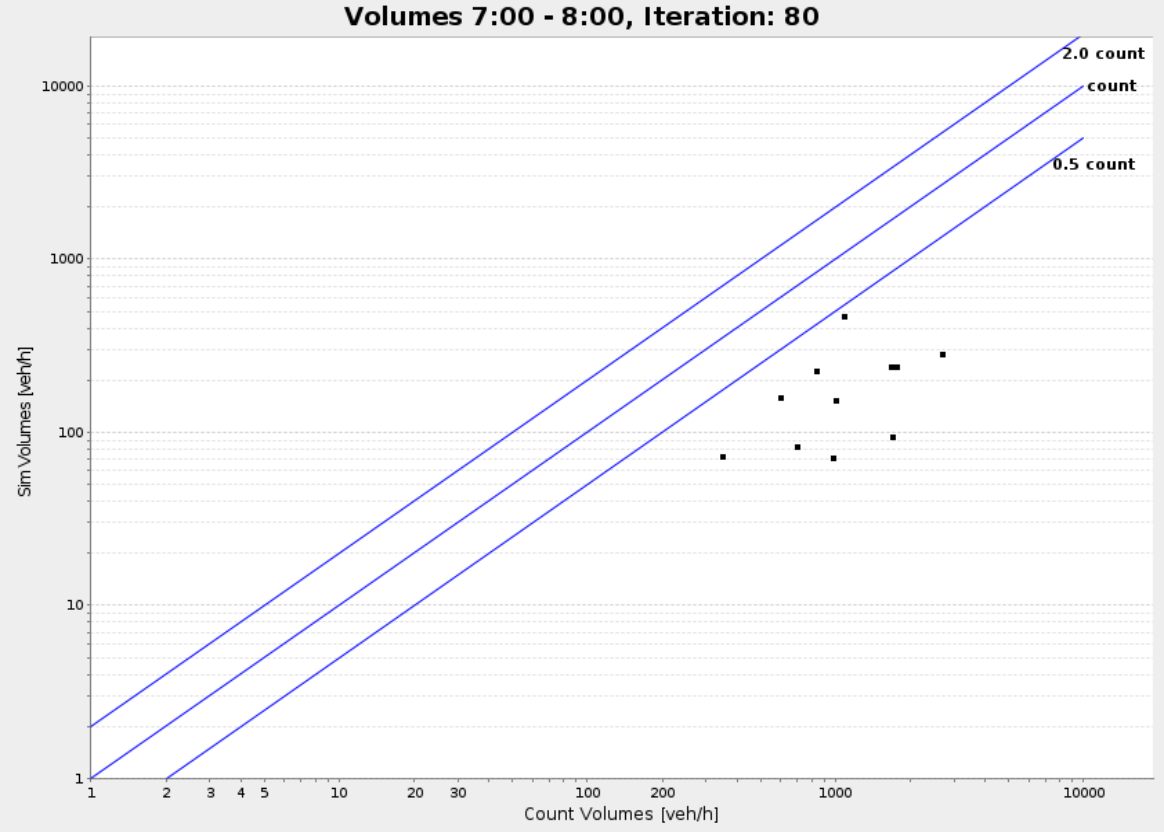


Figura 9.14: Conteos de 7-8 AM para el escenario 4

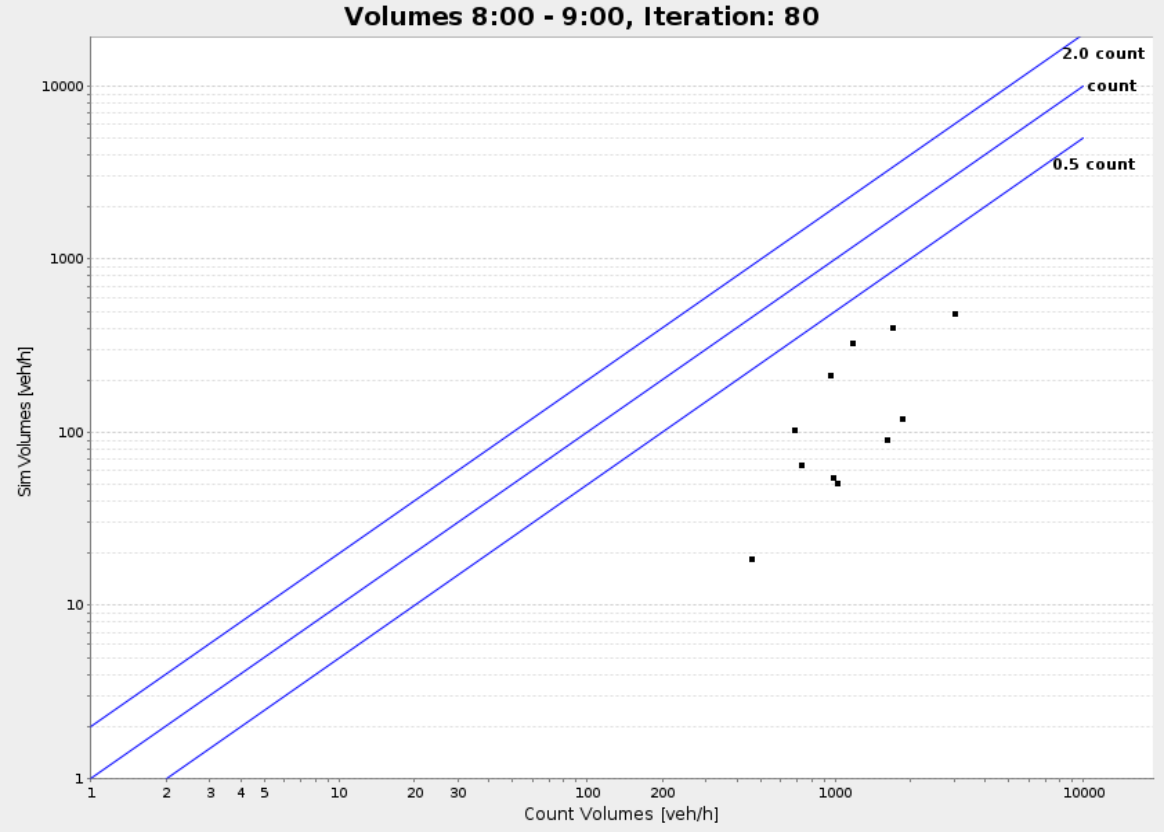


Figura 9.15: Conteos de 8-9 AM para el escenario 4

9.3.3.4. Volúmenes asignados a los enlaces en la última iteración

Enlace	Flujo obtenido	Flujo obtenido por URVISA	Diferencia Absoluta	Diferencia Relativa
1	2144	1027	1117	1,09
2	1941	1953	−12	−0,01
3	1503	2430	−927	−0,38
4	1974	1832	142	0,08
5	1347	2857	−1510	−0,53
6	1248	1649	−401	−0,24
7	1461	1247	214	0,17
8	958	1285	−327	−0,25
9	156	612	−456	−0,75
10	123	1051	−928	−0,88
11	334	992	−658	−0,66
12	1957	1759	198	0,11
13	4211	1962	2249	1,15
14	2232	1504	728	0,48
15	4211	2257	1954	0,87
16	87	747	−660	−0,88
17	1800	1471	329	0,22
18	988	1481	−493	−0,33
19	5600	6300	−700	−0,11
20	3657	5736	−2079	−0,36
21	1820	1486	334	0,22
22	1029	309	720	2,33
23	2013	1457	556	0,38
24	2942	1990	952	0,48
25	45	719	−674	−0,94
			Prom. Dif. Absoluta	772,72
			Prom. Dif. Relativa	0,56

Tabla 9.6: Tabla comparativa de flujos para el escenario 4

9.3.3.5. Tiempo de ejecución

Los resutados obtenidos por el comando *time*, fueron los siguientes:

```
real 57m34.402s
user 57m36.984s
sys 0m17.349s
```

9.3.3.6. Análisis de los resultados obtenidos

Para este escenario, el segundo con estrategia pura la cual consiste en el cambio del momento de salida, se obtienen resultados en la puntuación de los planes bastante paupérrimos y la razón de ello se puede observar en el histograma de eventos, donde se ve cómo muchos agentes llegan a su lugar de trabajo en horas vespertinas e incluso, nocturnas. Si bien los flujos obtenidos por enlace son bastante cercanos a los obtenidos en el estudio de URVISA y los conteos son buenos para la primera hora de la mañana, este escenario no es realista en lo absoluto, pues obliga a muchos agentes a llegar excesivamente tarde a sus puestos de trabajo. Por otro lado, este escenario permite demostrar que las vías de tiempo de viaje más corto (fundamentalmente las vías principales) no poseen la capacidad adecuada para suplir la demanda de tráfico en la hora pico de la mañana.

El tiempo promedio de viaje en este escenario fue de 02h:24m:02s y la distancia promedio recorrida por cada agente fue 1420.47 metros.

9.3.4. Escenario 6

9.3.4.1. Puntuación de los planes

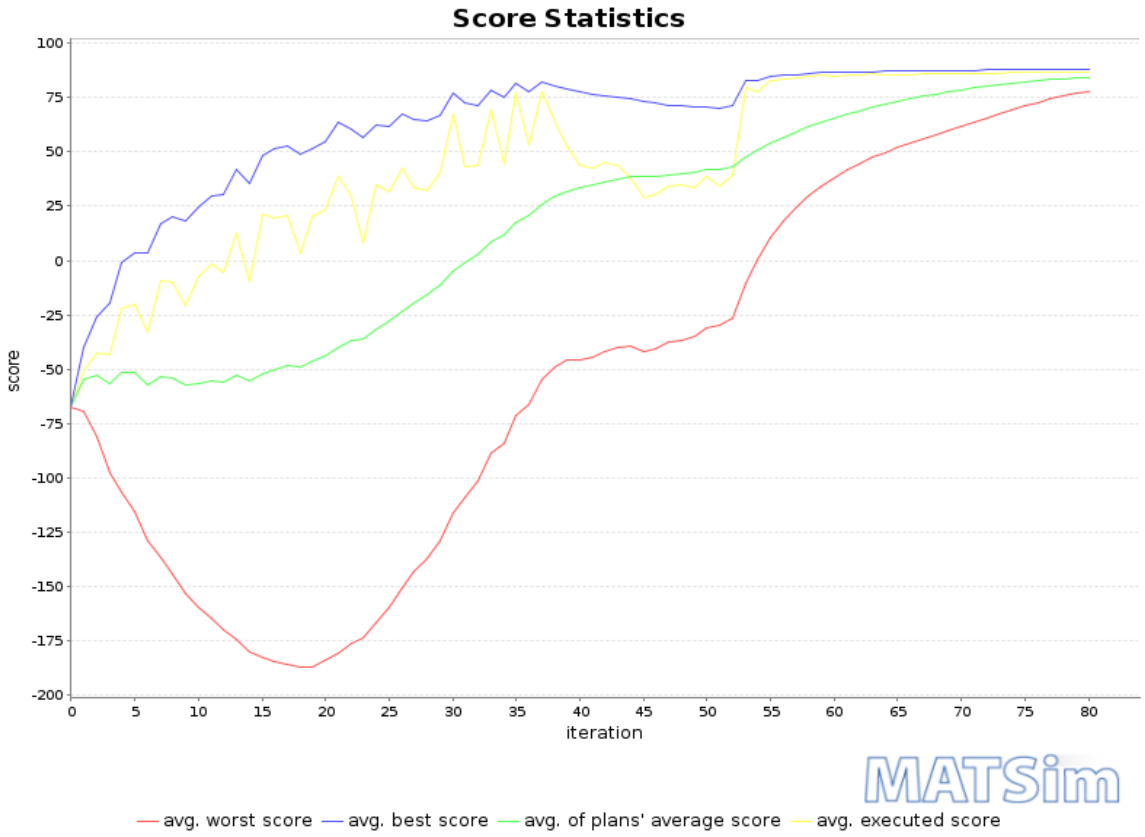


Figura 9.16: Puntación de los planes para el escenario 6

9.3.4.2. Histograma de eventos de la última iteración

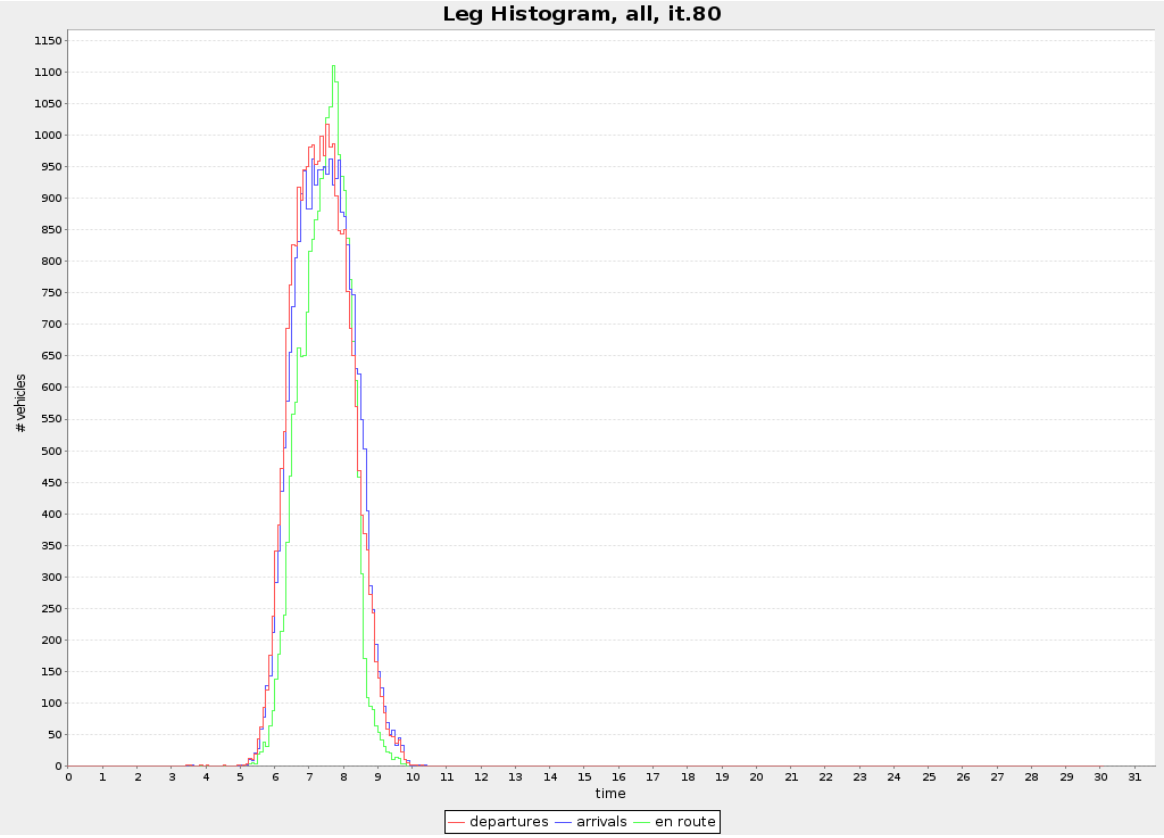


Figura 9.17: Histograma de eventos para el escenario 6

9.3.4.3. Conteos en las horas comprendidas entre 6 y 9 AM en la última iteración

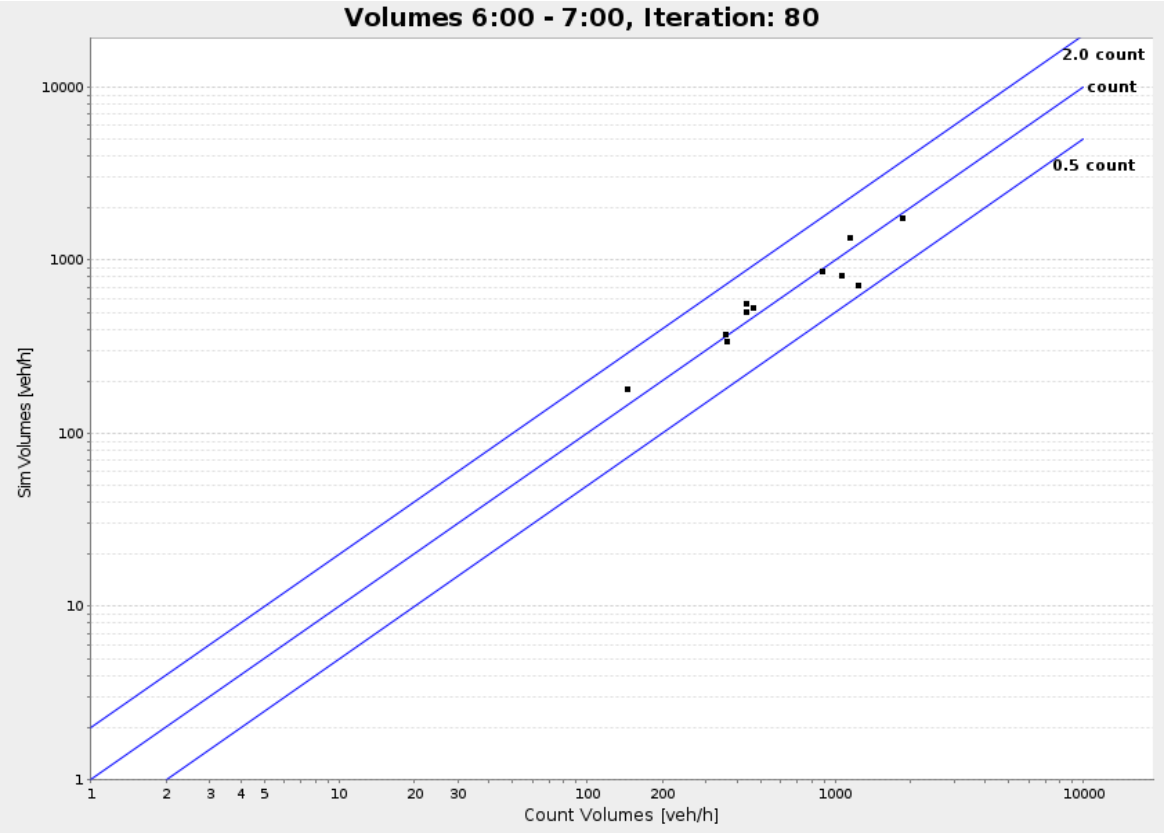


Figura 9.18: Conteos de 6-7 AM para el escenario 6

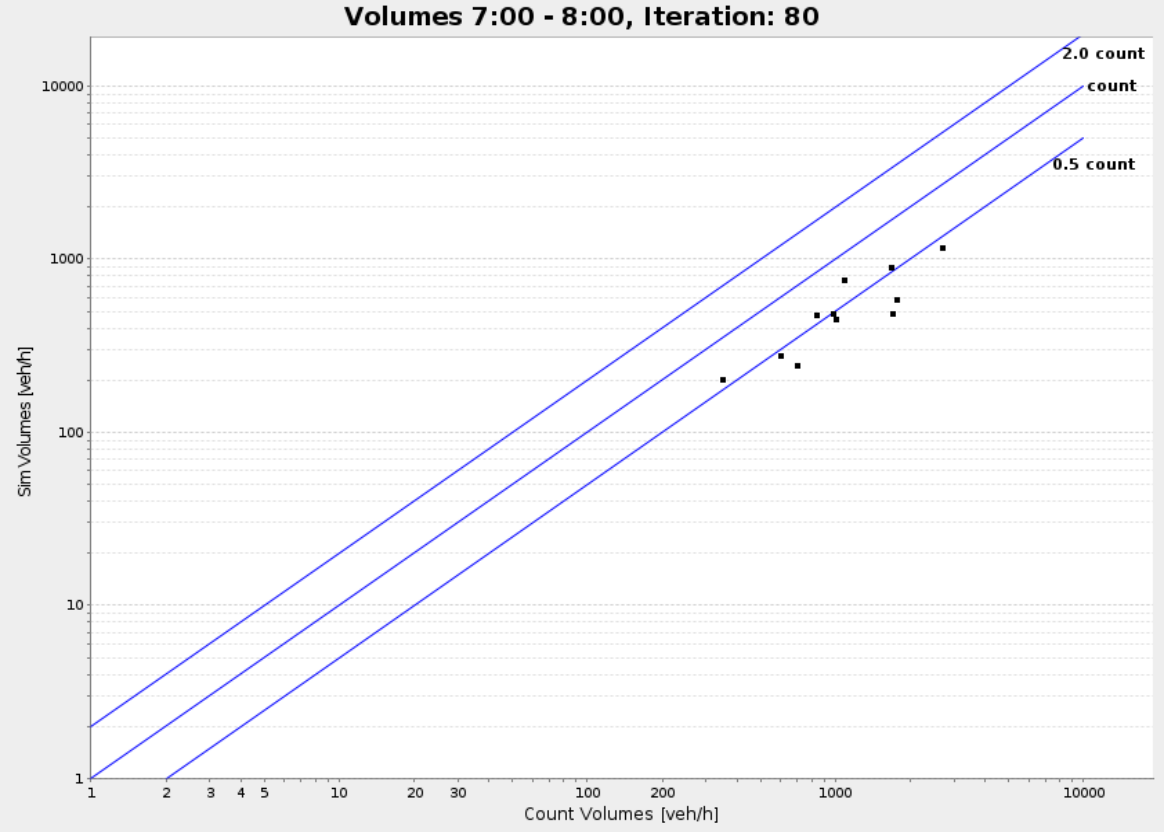


Figura 9.19: Conteos de 7-8 AM para el escenario 6

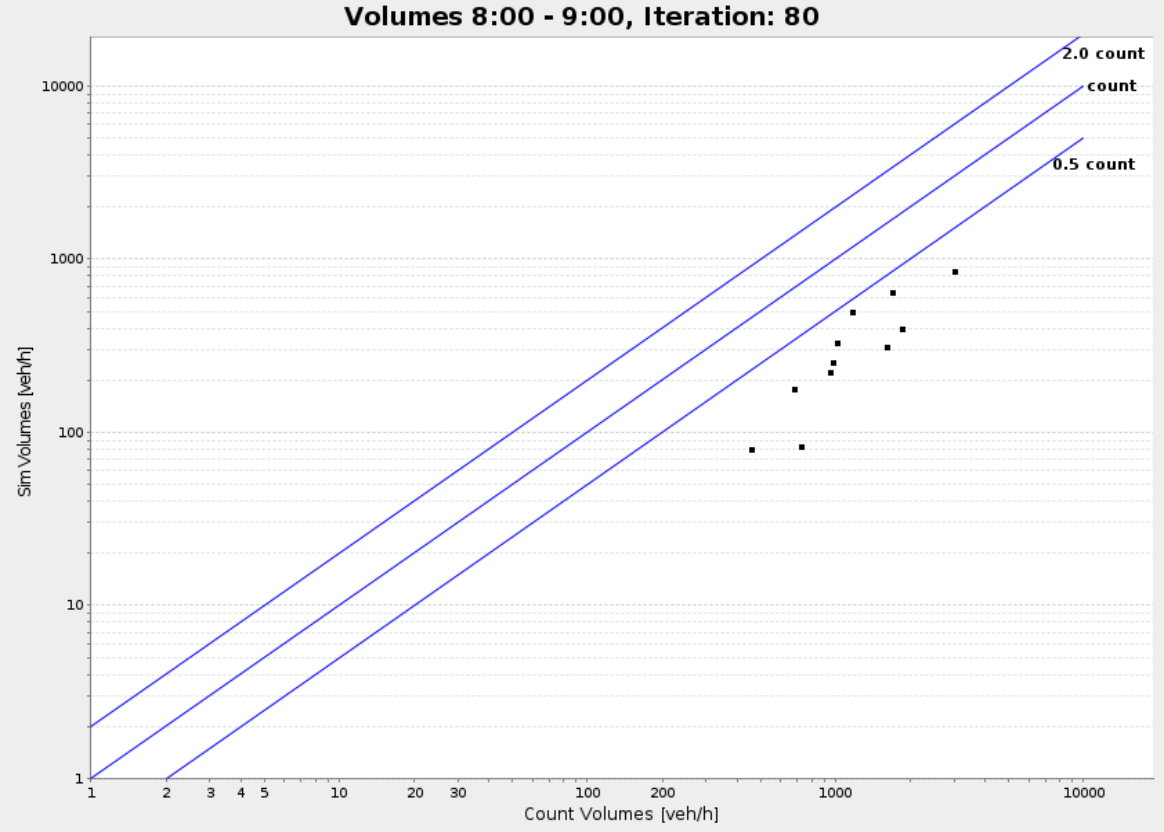


Figura 9.20: Conteos de 8-9 AM para el escenario 6

9.3.4.4. Volúmenes asignados a los enlaces en la última iteración

Enlace	Flujo obtenido	Flujo obtenido por URVISA	Diferencia Absoluta	Diferencia Relativa
1	2511	1027	1484	1,44
2	2120	1953	167	0,09
3	1812	2430	−618	−0,25
4	1654	1832	−178	−0,10
5	1658	2857	−1199	−0,42
6	1287	1649	−362	−0,22
7	1556	1247	309	0,25
8	1604	1285	319	0,25
9	430	612	−182	−0,30
10	954	1051	−97	−0,09
11	541	992	−451	−0,45
12	2380	1759	621	0,35
13	4821	1962	2859	1,46
14	3099	1504	1595	1,06
15	4821	2257	2564	1,14
16	994	747	247	0,33
17	2082	1471	611	0,42
18	1538	1481	57	0,04
19	5209	6300	−1091	−0,17
20	4411	5736	−1325	−0,23
21	1697	1486	211	0,14
22	1087	309	778	2,52
23	2013	1457	556	0,38
24	2970	1990	980	0,49
25	638	719	−81	−0,11
			Prom. Dif. Absoluta	757,68
			Prom. Dif. Relativa	0,51

Tabla 9.7: Tabla comparativa de flujos para el escenario 6

9.3.4.5. Tiempo de ejecución

Los resutados obtenidos por el comando *time*, fueron los siguientes:

```
real 26m48.510s
user 27m22.111s
sys 0m10.245s
```

9.3.4.6. Análisis de los resultados obtenidos

En este escenario los agentes usan una estrategia mixta que combina cambio de ruta con cambio de la hora de salida en una proporción de 1 a 3. Este escenario obtuvo la mejor puntuación de los planes de entre todos los escenarios, al igual que un histograma con una forma bastante aceptable. Por otro lado, los conteos obtenidos por este escenario son los que más se acercan a los conteos reales, verificando que no sólo es el escenario con mejor puntuación, sino el que más se acerca al tráfico vehicular real en la zona en las horas de la mañana. Los flujos obtenidos por enlace, si bien no son los que más se acercan a los obtenidos en el estudio de URVISA de entre todos los escenarios, siguen siendo muy semejantes en la mitad de los casos. Una posible conclusión derivada de este escenario, es que más que el cambio de ruta, los agentes deben optar por modificar la hora de salida de su hogar, pues cualquier ruta alterna que tomen se encontrará colmada de vehículos en la hora pico. Esta observación sigue siendo realista, pues el intenso tráfico de la ciudad de Caracas en la hora pico de la mañana ha obligado a muchas personas a salir mucho más temprano de sus hogares para evitar el congestionamiento y poder llegar a tiempo a sus lugares de trabajo o estudio.

El tiempo promedio de viaje en este escenario fue de 00h:04m:07s y la distancia promedio recorrida por cada agente fue 1645.09 metros.

Capítulo 10

Conclusiones y trabajos futuros

10.1. Conclusiones

El objetivo de MATSim es simular el tráfico vehicular. Para esto, requiere información previa del escenario a simular. El origen de estos escenarios es irrelevante para MATSim, por ello, estos escenarios deben ser creados, modificados y calibrados apoyados herramientas que no pertenecen al *toolkit* de MATSim.

La hetereogeneidad de los datos de entrada al simulador y las diferentes metodologías de trabajo de analistas y/o planificadores crean la necesidad del desarrollo de herramientas muy específicas que faciliten y hagan posible el manejo (creación, edición, calibración, etc) de esos datos. Para el caso de este trabajo, las herramientas específicas que se desarrollaron se enfocaron en la calibración de la red y la generación de la demanda de tráfico a partir de matrices origen destino, volviendose su uso indispensable para lograr los objetivos propuestos.

Dado que la metodología utilizada por URVISA para la generación y distrubución de demanda, que está basada en información agregada, difiere considerablemente del concepto propuesto por MATSim, donde se procura utilizar información desagregada (i.e. planes individuales por agentes) una comparación directa entre los resultados obtenidos por este trabajo y el realizado por URVISA carece de sentido estricto. Sin embargo, los resultados numéricos obtenidos por este trabajo son similares a los obte-

nidos por el trabajo de URVISA en cuanto a los volumenes comparados con los conteos y por ende, las observaciones asociadas a los resultados son similares.

10.2. Trabajos futuros

Fundamentado en todo el desarrollo de este Trabajo Especial de Grado, se proponen los siguientes trabajos futuros:

- Desarrollar y poner en práctica una metodología de recolección de datos desagregados para generar planes directamente a partir de censos, encuestas, etc, y no de matrices origen destino.
- Refinar la herramienta de creación de redes, como por ejemplo, facilitar el proceso de creación de enlaces de acuerdo a las características normativas de este.
- El desarollo de una herramienta que ayude a gestionar proyectos de simulación inspirada en la especificación formal de las “carpetas de proyecto” utilizada en este trabajo.

Apéndice A

Archivo XML de configuración de la red del ejemplo *equil* en MATSim

```
1 <?xml version="1.0" ?>
2 <!DOCTYPE config SYSTEM "http://www.matsim.org/files/dtd/config_v1.dtd">
3 <config>
4
5   <module name="global">
6     <param name="randomSeed" value="4711" />
7     <param name="coordinateSystem" value="Atlantis" />
8   </module>
9
10  <module name="network">
11    <param name="inputNetworkFile" value="examples/equil/network.xml" />
12  </module>
13
14  <module name="plans">
15    <param name="inputPlansFile" value="examples/equil/plans100.xml" />
16  </module>
17
18  <module name="controler">
19    <param name="outputDirectory" value="./output/equil" />
20    <param name="firstIteration" value="0" />
21    <param name="lastIteration" value="10" />
22    <param name="mobsim" value="queueSimulation" />
23  </module>
```

```
24
25  <module name="simulation">
26    <param name="startTime" value="00:00:00" />
27    <param name="endTime" value="00:00:00" />
28
29    <param name = "snapshotperiod"  value = "00:01:00"/>
30    <param name = "snapshotFormat"  value = "otfvis"/>
31  </module>
32
33  <module name="planCalcScore">
34    <param name="learningRate" value="1.0" />
35    <param name="BrainExpBeta" value="2.0" />
36
37    <param name="lateArrival" value="-18" />
38    <param name="earlyDeparture" value="-0" />
39    <param name="performing" value="+6" />
40    <param name="traveling" value="-6" />
41    <param name="waiting" value="-0" />
42
43    <param name="activityType_0"              value="h" /> <!-- home -->
44    <param name="activityPriority_0"          value="1" />
45    <param name="activityTypicalDuration_0" value="12:00:00" />
46    <param name="activityMinimalDuration_0" value="08:00:00" />
47
48    <param name="activityType_1"              value="w" /> <!-- work -->
49    <param name="activityPriority_1"          value="1" />
50    <param name="activityTypicalDuration_1" value="08:00:00" />
51    <param name="activityMinimalDuration_1" value="06:00:00" />
52    <param name="activityOpeningTime_1"       value="07:00:00" />
53    <param name="activityLatestStartTime_1" value="09:00:00" />
54    <param name="activityEarliestEndTime_1" value="" />
55    <param name="activityClosingTime_1"       value="18:00:00" />
56  </module>
57
58  <module name="strategy">
```

```
59      <param name="maxAgentPlanMemorySize" value="5" />
60
61      <param name="ModuleProbability_1" value="0.9" />
62      <param name="Module_1" value="BestScore" />
63
64      <param name="ModuleProbability_2" value="0.1" />
65      <param name="Module_2" value="ReRoute" />
66    </module>
67
68 </config>
```

Apéndice B

Archivo XML de los planes del ejemplo *equil* en MATSim

```
1 <?xml version="1.0" ?>
2 <!DOCTYPE plans SYSTEM "http://www.matsim.org/files/dtd/plans_v4.dtd">
3 <plans>
4 <person id="3">
5   <plan type="pt">
6     <act type="h" x="-25000" y="0" link="1" end_time="06:42:02" />
7     <leg mode="pt">
8     </leg>
9     <act type="w" x="10000" y="0" link="20" dur="09:02:22" />
10    <leg mode="pt">
11    </leg>
12    <act type="h" x="-25000" y="0" link="1" />
13  </plan>
14 </person>
15
16 <person id="4">
17  <plan type="car">
18 <!--    <act type="h" x="-25000" y="0" link="1" end_time="06:45" /> -->
19    <act type="h" x="-25000" y="0" link="1" end_time="05:58:30" />
20    <leg mode="car">
21      <route>2 7 12</route>
22    </leg>
23 <!--    <act type="w" x="10000" y="0" link="20" dur="9:14:24" /> -->
```

```
24      <act type="w" x="10000" y="0" link="20" dur="09:59:00" />
25      <leg mode="car">
26        <route>13 14 15 1</route>
27      </leg>
28      <act type="h" x="-25000" y="0" link="1" />
29    </plan>
30 </person>
31 </plans>
```

Apéndice C

Archivo XML la red del ejemplo *equil* en MATSim

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <!DOCTYPE network SYSTEM "http://www.matsim.org/files/dtd/network_v1.dtd
      ">
3
4 <network name="equil test network">
5  <nodes>
6   <node id="1" x="-20000" y="0" />
7   <node id="2" x="-15000" y="0" />
8   <node id="3" x="-865" y="5925" />
9   <node id="4" x="-2498" y="4331" />
10  <node id="5" x="-3829" y="3215" />
11  <node id="6" x="-4698" y="1711" />
12  <node id="7" x="-5000" y="0" />
13  <node id="8" x="-4698" y="-1711" />
14  <node id="9" x="-3829" y="-3215" />
15  <node id="10" x="-2498" y="-4331" />
16  <node id="11" x="-865" y="-5925" />
17  <node id="12" x="0" y="0" />
18  <node id="13" x="5000" y="0" />
19  <node id="14" x="5000" y="-10000" />
20  <node id="15" x="-20000" y="-10000" />
21 </nodes>
22 <links capperiod="01:00:00">
```

```
23  <link id="1" from="1" to="2" length="10000.00" capacity="36000"
        freespeed="27.78" permlanes="1" />
24  <link id="2" from="2" to="3" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
25  <link id="3" from="2" to="4" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
26  <link id="4" from="2" to="5" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
27  <link id="5" from="2" to="6" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
28  <link id="6" from="2" to="7" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
29  <link id="7" from="2" to="8" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
30  <link id="8" from="2" to="9" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
31  <link id="9" from="2" to="10" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
32  <link id="10" from="2" to="11" length="10000.00" capacity="3600"
        freespeed="27.78" permlanes="1" />
33  <link id="11" from="3" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
34  <link id="12" from="4" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
35  <link id="13" from="5" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
36  <link id="14" from="6" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
37  <link id="15" from="7" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
38  <link id="16" from="8" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
39  <link id="17" from="9" to="12" length="5000.00" capacity="1000"
        freespeed="27.78" permlanes="1" />
```

```
40   <link id="18" from="10" to="12" length="5000.00" capacity="1000"
         freespeed="27.78" permlanes="1" />
41   <link id="19" from="11" to="12" length="5000.00" capacity="1000"
         freespeed="27.78" permlanes="1" />
42   <link id="20" from="12" to="13" length="10000.00" capacity="36000"
         freespeed="27.78" permlanes="1" />
43   <link id="21" from="13" to="14" length="10000.00" capacity="36000"
         freespeed="27.78" permlanes="1" />
44   <link id="22" from="14" to="15" length="35000.00" capacity="36000"
         freespeed="27.78" permlanes="1" />
45   <link id="23" from="15" to="1" length="10000.00" capacity="36000"
         freespeed="27.78" permlanes="1" />
46 </links>
47 </network>
```

Apéndice D

Definición de las funciones públicas de la interfaz *Network* de MATSim

NetworkFactory getFactory() : retorna el constructor de los elementos de la red en la forma de un NetworkFactory.

Map<Id, ? extends Node> getNodes() : retorna un mapa que contiene a todos los nodos de la red.

Map<Id, ? extends Link> getLinks() : retorna un mapa que contiene a todos los enlaces de la red.

void addNode(Node n) : agrega un nuevo nodo a la red.

void addLink(Link l) : agrega un nuevo enlace a la red.

Node removeNode(final nodeId) : elimina un nodo de la red. Retornará el nodo eliminado. En caso de no existir tal Id en la colección de nodos, entonces se retornará *null*.

void removeNode(final nodeId) : elimina un enlace de la red. Retornará el enlace eliminado.En caso de no existir tal Id en la colección de enlaces, entonces se retornará *null*.

Apéndice E

Definición de las funciones públicas de la interfaz *Node* de MATSim

boolean addInLink(Link link) : agrega un nuevo enlace de entrada

boolean addOutLink(Link link) : agrega un nuevo enlace de salida.

Map<Id, ? extends Link> getInLinks() : devuelve todos los enlaces de entrada asociados al nodo en la forma de un mapa.

Map<Id, ? extends Link> getOutLinks() : devuelve todos los enlaces de salida asociados al nodo en la forma de un mapa.

Apéndice F

Definición de las funciones públicas de la interfaz *Link* de MATSim

boolean setFromNode(Node node) : establece el nodo donde comienza el enlace.

boolean setToNode(Node node) : establece el nodo donde termina el enlace.

void setFreespeed(double freespeed) : establece la velocidad de flujo libre del enlace.

void setLenght(double lenght) : establece la longitud del enlace.

void setNumberOfLanes(double lanes) : establece la cantidad de canales del enlace.

void setCapacity(double capacity) : establece la capacidad del enlace.

void setAllowedModes(Set< String > modes) : establece los modos permitidos en el enlace, i.e., vehícular, peatonal, etc.

Cada uno de los métodos anteriormente descritos poseen sus métodos get() asociados.

Apéndice G

Salida del simulador: histograma de eventos

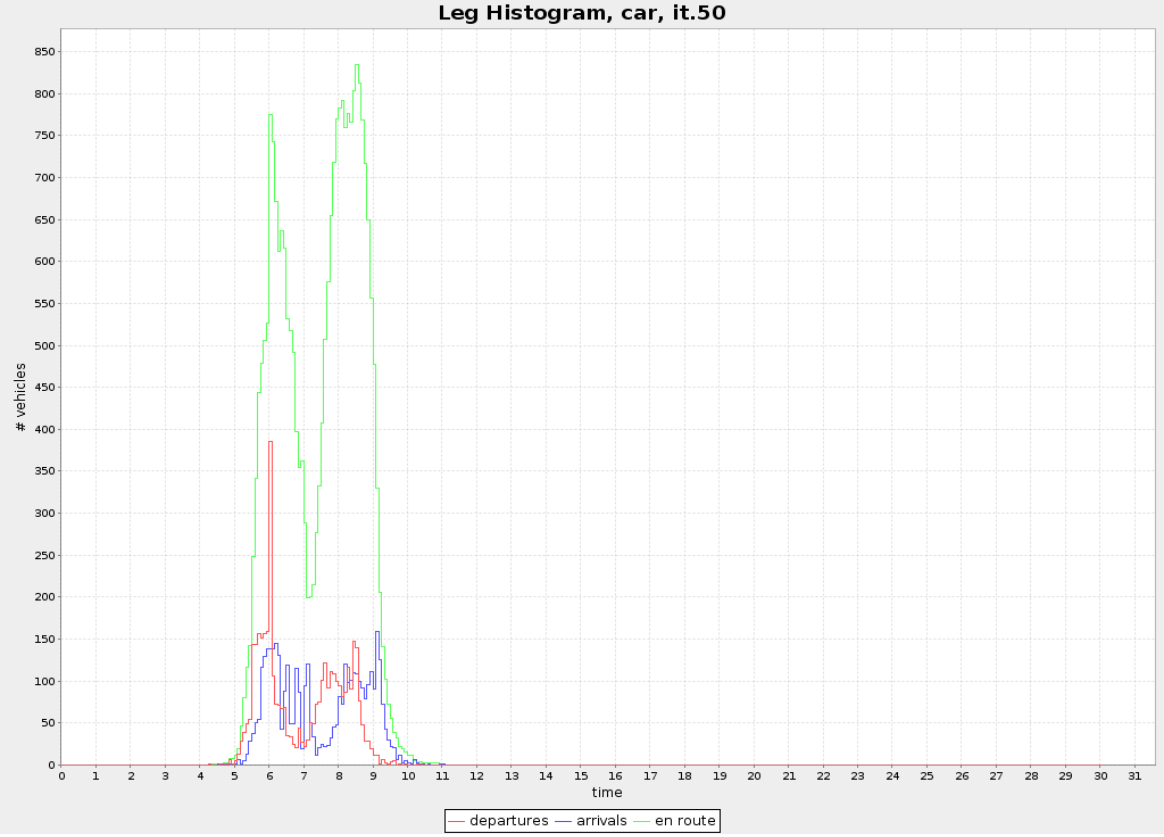


Figura G.1: Histograma de eventos

Apéndice H

Salida del simulador: puntuación de los planes ejecutados

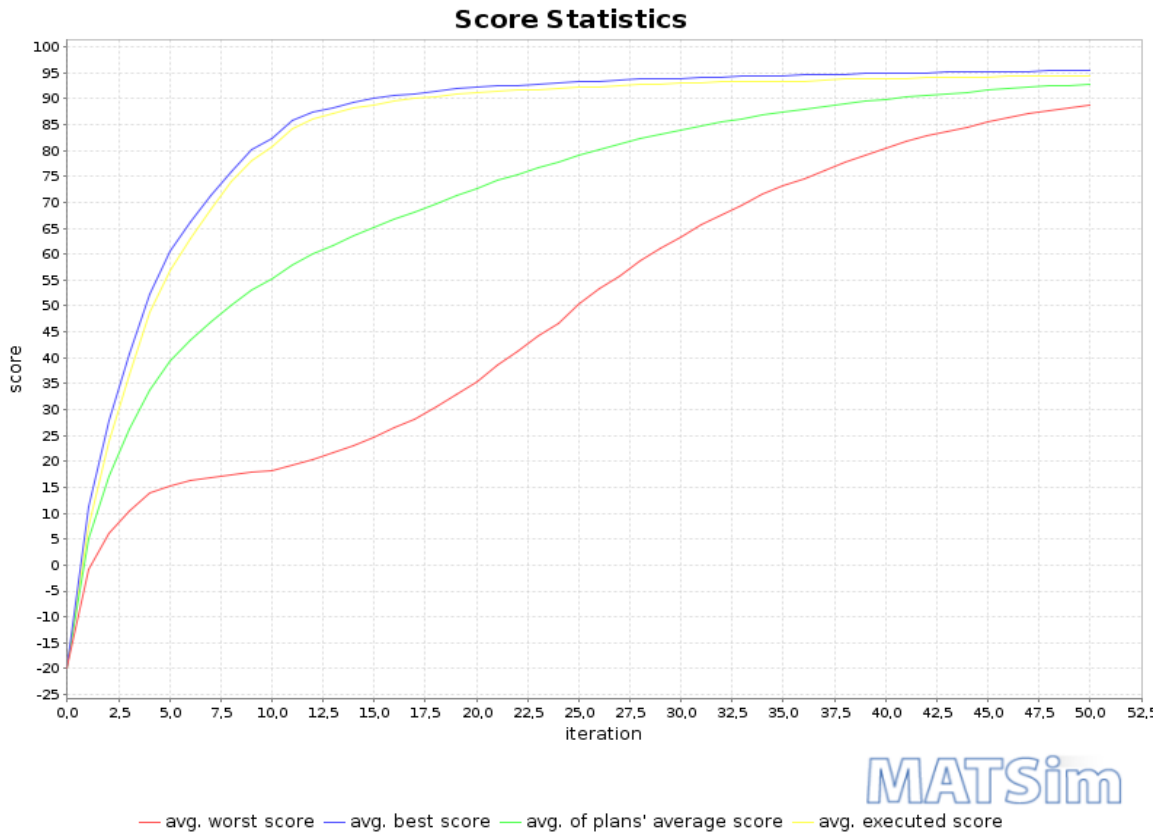


Figura H.1: Puntuación de los planes ejecutados

Apéndice I

Información de generación de tráfico

Sector	Área (m^2)	Vehículos	Viajes	Índice de viajes por m^2	Origen	Destino
1	26855,48	0	0	3,9	99	552
2	58506,46	838	5,1	12,62	327	1515
3	11728,4	82	2,82	7,68	95	228
4	31886,59	243	2,58	4,78	118	427
5	19678,24	93	3,64	11,13	37	208
6	20802,55	89	0,99	5,29	34	150
7	0	0	0	0	0	0
8	30537,48	20	0,42	1,6	35	226
9	37091,88	283	3,14	4,8	121	738
10	8432,43	34	1,82	2,62	46	87
11	23504,4	0	0	8,1	59	257
12	42980,7	111	1,59	12,99	68	353
13	67574,48	0	0	14,7	204	1499
14	43559,59	376	1	1,42	60	401
15	31880,6	28	0,35	8,75	73	529
16	21828	0	0	5	42	308
17	32852,79	0	0	2,9	53	386
18	14034,47	0	0	0,8	15	112
19	5679,8	0	0	2,23	19	52
20	49100,4	0	0	4,2	140	1031

21	15054,22	0	0	3,7	52	192
22	27622,86	0	0	0,1	10	30
23	0	0	0	0	14	44
24	0	0	0	0	12	60
25	64727,5	74	2,48	10,38	155	1135
26	11748,13	353	9,94	10,74	296	390
27	60141,48	0	0	6,52	93	1009
28	16535,6	0	0	0,8	18	132
29	159893,6	0	0	1,6	110	232
30	26520	0	0	0,8	29	212
31	3570,23	0	0	0,8	4	29
32	21421,37	0	0	0	23	171
33	23887,85	618	7,28	9,38	104	757
34	13197,71	0	0	1,6	14	106
35	24011,6	0	0	4,7	139	354
36	23206,76	55	0,8	11,5	51	245
37	8694	0	0	3,2	10	70
38	68088,58	161	0,47	3,9	455	876
39	19440	0	0	0,6	161	117
40	21763,25	0	0	1,3	39	283
41	34513,56	0	0	1,3	61	449
42	13294,87	0	0	1,3	24	173
43	64045,76	0	0	1,3	114	833
44	28760,7	0	0	1,3	51	374
45	30041,33	0	0	1,3	53	391
46	24675,41	0	0	1,3	44	321
47	4588,35	0	0	1,3	8	60
48	18873,88	0	0	0,8	21	151
49	4819,2	0	0	0,8	11	39

Apéndice J

Matriz origen-destino generada por URVISA

Zonas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	60	0	19	36	117	92	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	35	54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	5	0	36	55	1	0	1	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	15	0	0	0	0	0	0	8	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	8	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	14	0	0	0	0	0	0	6	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	24	0	0	0	0	0	0	9	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	12	101	17	16	3	12	0	10	59	2	10	14	63	54	25	15	6	4	2	26	4	1	3	6	42	32	49	74	18	12	11	4	13	34	5	10	31	5	9	9	51	66	25	110	49	75	45	31	5	58	443	356	0	181	0	0	0	115	0
51	21	134	25	28	8	19	0	15	73	5	18	19	88	73	34	20	24	5	5	44	10	4	10	8	72	49	17	18	12	11	4	1	3	14	1	3	10	3	3	0	44	57	17	84	36	50	31	5	25	0	309	463	0	0	65	0	0	0	351
52	9	50	10	10	3	5	0	4	30	2	7	7	31	21	15	8	5	1	1	10	4	1	2	2	25	17	12	5	11	4	1	3	14	1	3	10	3	3	0	3	66	25	17	84	18	45	10	8	0	0	0	0	0	0	0	0	0	0	23
53	48	266	54	52	17	31	0	26	125	9	35	40	139	106	70	45	24	8	8	72	13	7	17	8	11	84	115	25	62	26	6	32	87	6	17	48	17	23	33	41	77	24	124	60	45	58	8	0	0	138	443	356	0	0	0	0	0	755	32
54	8	36	3	6	0	5	0	3	16	1	5	4	21	12	7	2	5	4	2	20	4	3	2	2	26	15	34	5	13	5	1	10	23	2	3	7	4	6	6	4	9	3	18	9	4	5	1	0	0	0	0	181	0	0	0	115	0	0	117
55	30	11	18	32	12	21	0	1	60	7	20	22	39	26	10	26	16	5	1	7	9	2	3	1	46	35	47	11	5	10	2	12	6	6	14	29	6	2	10	11	22	6	37	14	11	15	2	0	0	0	0	45	148	0	39	0	0	0	161
56	172	334	22	69	16	33	0	24	73	12	34	33	60	43	132	106	16	5	6	40	9	2	3	6	47	35	46	12	25	7	4	11	33	6	15	42	6	8	6	25	47	16	73	39	28	36	6	19	0	0	133	0	0	0	175	96	0	0	146
57	105	285	21	60	15	32	0	24	73	12	34	34	60	48	33	16	7	6	2	40	9	2	3	6	47	35	46	12	25	7	4	11	33	6	15	42	6	8	6	25	47	16	73	39	28	36	6	19	0	0	133	0	0	0	175	96	0	0	97
58	46	13	43	53	14	28	0	22	147	10	34	34	48	32	54	34	33	7	7	68	87	7	11	17	146	76	111	29	48	16	5	27	35	8	15	11	11	4	18	40	68	51	235	98	70	49	19	38	0	110	97	0	858	78	0	0	0	0	77

Apéndice K

Volúmenes obtenidos por enlace en el estudio realizado por URVISA

Bibliografía

[1] The American Heritage Dictionary of the English Language, 4 Ed. Boston: Houghton Mifflin Company, 2000.

[2] Drake, Miriam A. Encyclopedia of Library and Information Science. s.l.: CRC Press, 2003.

[3] Staines, G.M., Bonacci, M and Jonhson, K. Social Sciences Research: Writing Strategies for Students. s.l.: Scarecrow Press: Lanham, 200.

[4] Smith, Roger D. Simulation. Encyclopedia of Computer Science. 4ta. Edición. New York: Nature Publishing Group, 2000.

[5] Sulistio, Anthony, Yeo, Chee Shin y Buyya, Rajkumar. A taxonomy of computer-based simulations and its mapping to parallel and distributed systems simulation tools. Melbourne, Australia: Department of Computer Science and Software Engineering, University of Melbourne, 2004.

[6] Volchan, Sérgio B. What Is a Random Sequence? s.l.: THE MATHEMATICAL ASSOCIATION OF AMERICA, 2002, The American Mathematical Monthly, pp. 46 - 63.

[7] Gentle, James E. Random Number Generation and Monte Carlo Methods. s.l.: Springer, 2003.

[8] Press, William H., et al. Numerical Recipes. New York: Cambridge University Press, 2007.

[9] Park, Steve y Miller, Keith. Random number generators: good ones are hard to find. New York: ACM, 1988.

[10] Blum, Manuel, Blum, Leonore y Shub, Michael. A Simple Unpredictable Pseudo-Random Number Generator. s.l.: SIAM Journal on Computing, 1986.

[11] Matsumoto, Makoto and Nishimura, Takuji. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. s.l. : ACM Transactions, 1998.

[12] Devroye, Luc. Non- Uniform Random Variate Generation. New York: Springer-Verlag, 1986.

[13] Box, George Edward Pelham y Muller, Mervin Edgar. A Note on the Generation of Random Normal Deviates. 1958, The Annals of Mathematical Statistics, Vol. Vol. 29, pp. 610–61.

[14] Box Muller Transformation. [En línea] [Citado: Enero 15, 2009.] http://es.wikipedia.org/wiki/Método_de_Box-Muller.

[15] Rusell, Stuart y Norvig, Peter. Inteligencia Artificial un enfoque moderno. México: Prentice Hall Hispanoamericana, 1996.

[16] Jennings, N. R. and Wooldridge, M. Applications of Intelligent Agents. Londres: University of London, 1998.

[17] Maes, Pattie. Artificial Life Meets Entertainment: Life like Autonomous Agents. s.l.: Communications of the ACM, 1995.

[18] Franklin, Stan y Graesser, Art. Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. Memphis: Springer-Verlag, 1996.

[19] Lesser, V. Multiagent Systems: An Emerging Subdiscipline of AI. s.l.: ACM Computing Surveys, 1995. pp. pp. 340-342. Vols. Volume 27, Number 3.

[20] Shoham, Yoav y Leyton-Brown, Kevin. Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundation. Cambrige: Cambrige University Press, 2008.

[21] Woolridge, Michael. An Introduction to MultiAgent Systems. West Sussex: John Wiley & Sons, LTD, 2002.

[22] Austin, John Langshaw. How to Do Things with Words. s.l.: Harvard University Press, 1962.

[23] Dirven, René y Verspoor, Marjolyn. Cognitive exploration of language and linguistics. Amsterdam: John Benjamins, 2004.

[24] Searle, John. Speech Acts. Cambridge: Cambridge University Press, 1969.

[25] Weiss, Gerhard. Multiagent systems: a modern approach to distribuited artificial intelligence. Boston: Massachusetts Institute of Technology, 1999.

[26] Helbing, Dirk. Traffic and Related Self-Driven Many-Particle Systems. 2001, Reviews of Modern Physics, pp. 1067-1141.

[27] Gartner, Nathan, Messer, Carroll J. and Rathi, Ajay K. Traffic Flow Theory. s.l.: Transportation Research Board, 1992.

[28] Gazis, Denos. Traffic Theory. New York: Kluwer Academics Publishers, 2000.

[29] Kesting, Arne, Treiber, Martin y Helbing, Dirk. Agents for Traffic Simulation. 2008, Multi-Agent Systems: Simulation and Applications, pp. 325-356.

[30] Rothery, Richard W. Car Following Models. s.l.: Transportation Research Board, 1992.

[31] Webster, Nathan, Suzuki, Takahiro y Kuwahara, Masao. Driver Model for Traffic Simulation, with Tactical Lane Changing Behavior. 3, 2007, SEISAN KENKYU, Vol. 59, pp. 188-191.

[32] Wolfram, Stephen. History of cellular automata. A new kind of Science. Londres: Wolfram Media, 2002, p. 876.

[33] Wolfram, Stephen. Cellular automata as models of complexity. 1984, Nature, Vol. 311, pp. 419-424.

[34] Nagel, Kai y Schreckenberg, Michael. A cellular automaton model for freeway traffic. París: Journal Physics, 1992.

[35] Helbing, Dirk, Hennecke, Ansgar y Treiber, Martin. Congested traffic states in empirical observations and microscopic simulations. 2000, Physical Review, pp. 1805–1824.

[36] Treiber, Martin y Helbing, Dirk. Memory effects in microscopic traffic models and wide scattering in flow-density data. 2003, Physical Review, Vol. 68.

[37] Brilon, W. y Ponzlet, M. Application of traffic flow models. Singapore: s.n., 1996. Traffic and Granular Flow. pp. 23-40.

[38] Hamdar, Samer H., et al. Modeling Driver Behavior as Sequential Risk-Taking Task. 2008, Transportation research record, pp. 208-217.

[39] Kahneman, Daniel y Tversky, Amos. Prospect Theory: An Analysis of Decision under Risk. 1979, Econometrica, pp. 263-291.

[40] Gipps, P. G. A model for the structure of lane-changing decisions. 5, 1986, Transportation Research Part B: Methodological, Vol. 20, pp. 403-414.

[41] Kesting, Arne, Treiber, Martin y Helbing, Dirk. General Lane-Changing Model MOBIL for Car-Following Models. 2007, Transportation research record, pp. 86-94.

[42] Cao, Jian, et al. Towards Building an Intelligent Traffic Simulation Platform. Singapore: IEEE, 2006. Cluster Computing and the Grid Workshops, 2006. Sixth IEEE International Symposium on. pp. 64-70.

[43] Chopra, Amit K. y Singh, Munindar P. An Architecture for Multiagent Systems: An Approach Based on Commitments. Mayo 2009, Proceedings of the AAMAS Workshop on Programming Multiagent Systems .

[44] Soler, José, et al. Towards a Real-Time Multi-Agent System Architecture. Barcelona: s.n., 2002.

[45] Flores-Mendez, Roberto A. Towards a Standardization of Multi-Agent System Frameworks. 16, s.l.: ACM, 2009, The ACM Student Journal: Crossroad.

[46] Rathi, Ajay K. y Santiago, Alberto. Urban Network Traffic Simulation:TRAF-NETSIM Program. 6, 1990, Journal of Transportation Engineering, Vol. 116, pp. 734-743.

[47] Miller, Jeffrey. FreeSim: A Free Real-Time Freeway Traffic Simulator. [En línea] [Citado: Febrero 9, 2009.] http://www.freewaysimulator.com/.

[48] McTrans. CORSIM: Microscopic Traffic Simulation Model. [En línea] [Citado: Febrero 9, 2009.] http://mctrans.ce.ufl.edu/featured/tsis/Version5/corsim.htm.

[49] Quadstone. Quadstone Paramics - Product Overview. [En línea] [Citado: Febrero 9, 2009.] http://www.paramics-online.com/products.php.

[50] Massive. Massive - Real World Simulations. [En línea] [Citado: Febrero 9, 2009.] http://www.massivesoftware.com/real-world-simulation/.

[51] SourceForge.net: sumo. [En línea] [Citado: Febrero 9, 2009.] http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Main_Page.

[52] Universidad Técnica de Berlín. Multi-Agent Transport Simulation — MATSim. [En línea] [Citado: Febrero 9, 2009.] http://matsim.org/.

[53] Brassesco, Javier. Un millón de vehículos están de más en las vías de la capital. El Universal. Septiembre Lunes 03, 2007, p. Caracas.

[54] Negrón, Marco. Transporte en Emergencia. El Universal. 19 de Octubre de 2008, pág. Opinión.

[55] Texas Transportation Institute. Texas Transportation Institute. [En línea] 2005. [Citado: Mayo 18, 2009.] http://mobility.tamu.edu/ums/congestion_data/tables/national/table_2.pdf.

[56] Goodwin, Phil. The Economic Cost of Road Traffic Congestion. Londres: Rail Freight Group, 2004.

[57] Cañizales V., Migdalis. La hora pico se volvió permanente. El Universal. 14 de Mayo de 2007, págs. 4-1.

[58] Wilson, Peter. Venezuela: Land of 12-Cent Gas. Businees Week. Mayo 23, 2008.

[59] El Universal. Pdvsa pierde 2,1 millardos por subsidio a la gasolina. El Universal. Abril 13, 2009, p. Economía.

[60] The Economist. The price of cheap petrol. The Economist. Septiembre 11, 2008.

[61] Solomon, S., D. Qin, M. Manning,Z. Chen, M. Marquis, K.B. Averyt, M.Tignor y H.L. Miller. IPCC, 2007: Summary for Policymakers. In: Climate Change 2007: The Physical Science Basis. Contribution of Working Group I to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change. Cambrige, UK. New York, USA: Cambridge University Press, 2007.

[62] World Resources Institute. CAIT. [En línea] 2005. [Citado: 05 18, 2009.] http://cait.wri.org/.

[63] Miller, Peter. Ahorro de Energía: Se empieza en casa. Marzo 2009, National Geographic, en Español, p. 1.

[64] Marsaglia, George y Tsang, Wai Wan. The Ziggurat Method for Generating Random Variables. 2000, Journal of Statistical Software.

[65] Cetin, N. y K. Nagel. Parallel queue model approach to traffic micro-simulations. Paper presentado en el 82th Annual Meeting of the Transportation Research Board, Washington, D.C., Enero 2003.

[66] C. Gawron. An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model. International Journal of Modern Physics C, , 1998.

[67] P. Chakroborty y A. Das. Principles of Transportation Engineering. Prentice Hall of India. Nueva Delhi, 2005.

[68] Cetin, N., Nagel, K. y Burri, A. A Large-Scale Agent-Based Traffic Microsimulation Based On Queue Model. 3rd. Swiss Transport Research Conference. Monte Verità, Ascona. Marzo 19-21, 2003.

[69] Wright, Paul. Highway Engineering. 6th ed. Wiley, 1996

[70] Transportation Planning Handbook. Institute of Transportation Engineers. 1992.

[71] Wardrop, John Glen. Some Theoretical Aspects of Road Traffic Research. Road Engineering Division Meeting, 1952.

[72] Correa, José y Stier-Mosses, Nicolas. Wardrop Equilibria. Encyclopedia of Operation Research and Management Science. 2010.

[73] IIT Bombay. Notas de clase. Otoño, 2006.

[74] Balmer, M., Axhausen, K. y Nagel, K. An Agent Based Demand Modeling Framework for Large Scale Micro-Simulations. 2005.

[75] Balmer, M. Travel Demand Modeling for Multi-Agent Transport Simulations: Algorithms and Systems. ETH Zurich, 2007.

[76] B. Ran y D. Boyce. Modeling Dynamic Transportation Networks. Springer, 2 edition, 1996.

[77] http://www.openstreetmap.org/

[78] http://es.wikipedia.org/wiki/Shapefile

[79] http://mercurial.selenic.com/about/

[80] Estudio de actualización de transporte y circulación de Los Cortijos de Lourdes y Los Ruices. Urbanismo y vialidad S.A. (URVISA). 2008.

[81] Song, X., Rudorfer, A., Hwong, B., Matos, G., Nelson, C. S-RaP: A concurrent, Evolutionary Software Prototyping Process. Siemens Corporate Research Inc. Springer-Verlag, Berlin. 2005.