Universidad de la Republica

PROYECTO DE GRADO

Ingenieria en computacion

Algoritmos evolutivos en sincronización de semáforos en el Corredor de Garzón

Alvaro Acuña
Efrain Arreche

2014

Supervisor: Sergio Nesmachnow

Centro de Calculo - Instituto de Computacion
Montevideo, Uruguay

Algoritmos evolutivos en sincronización de semáforos en el Corredor de Garzón
Acuña, Alvaro - Arreche, Efrain
Proyecto de Grado
Instituto de Computación - Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay, diciembre de 2014

Algoritmos evolutivos en sincronización de semáforos en el Corredor de Garzón

Resumen

El proyecto propone el estudio de la sincronización de semáforos como problema de optimización multiobjetivo, y el diseño e implementación de un algoritmo evolutivo para resolver el problema con alta eficacia numérica y desempeño computacional en un escenario real utilizando simulaciones del tráfico.

Se toma como aplicación la sincronización de semáforos del "Corredor de Garzón" (Montevideo, Uruguay) dado que la cantidad de cruces, calles, tráfico y cantidad de semáforos lo hace un problema interesante desde el punto de vista de su complejidad.además es real y admitido por las autoridades responsables los problemas en este sentido por lo que todavía hay espacio para la mejora de los tiempos promedio de los viajes.

El problema de sincronización de semáforos es NP-difícil y no existe (hasta el momento) un método determinístico que lo resuelva, se buscará mediante un algoritmo evolutivo llegar a una configuración aceptable de los semáforos minimizando los tiempos de espera de los vehículos y mejorando de esta manera la calidad del tráfico. Buscamos demostrar que estas técnicas son aplicables a escenarios reales y que deberían considerarse en el abanico de posibilidades disponibles.

Palabras clave: Algoritmo Evolutivo, Sincronizacion semáforos, escenario real, Cluster

Índice general

Índice de figuras

Índice de cuadros

Índice de Algoritmos

Capítulo 1

Introducción

He llamado a este principio, por el cual cada pequeña variación, si útil, es preservada, con el término de Selección Natural
— Charles Darwin, El origen de las especies

En esta sección se pretende introducir al lector en el contexto general donde se desarrolla este trabajo así como los objetivos buscados.

1.0.1. Motivación y contexto

Los algoritmos evolutivos han demostrado su utilidad en problemas complejos y particularmente uno de ellos es la sincronización de semáforos. Se busca desarrollar un algoritmo que logre resolver este problema con buenas métricas. Por la flexibilidad inherente de este algoritmo es que no está destinado a resolver el problema en una zona en particular sino que se podría aplicar en forma general.

En este sentido se eligió la zona del corredor Garzón que presenta particularidades que la destacan y la hacen interesante desde el punto de vista de la investigación. Su complejidad viene dado por el largo del tramo, la cantidad de cruces, la complejidad y cantidad de semáforos en cada uno de ellos, las distintas reglas de tráfico aplicadas a cada tramo como por ejemplo exclusividad del ómnibus, o distinción para doblar a la izquierda, tráfico vehicular y transporte publico, calles no paralelas, entre otros.

Por tanto al probar que el algoritmo funciona en esta zona tan compleja se puede tener confianza de que se comportaría adecuadamente en otras zonas que no presentan tanta complejidad y obtener buenos resultados.

1.0.2. Objetivos

Estos son los objetivos básicos que se plantearon al inicio del proyecto.

- Estudio del problema del tráfico y la sincronización de semáforos.
- Relevamiento de información sobre trabajos relacionados en este ámbito.
- Creación de un algoritmo evolutivo que resuelva el problema en la zona del corredor Garzón.
- Confección de el mapa y la configuración relativa a semáforos, tráfico y reglas de transito.
- Demostrar que los algoritmos evolutivos pueden solucionar problemas complejos en escenarios reales, siendo una herramienta perfectamente utilizable.
- Aplicar técnicas de computación de alto desempeño para aumentar el rendimiento de la solución.
- Probar la escalabilidad de la solución.

1.0.3. Enfoque

Desde un primero momento se intento dotar al proyecto de una buena aproximación de la realidad, en tal sentido se realizaron reuniones con el Ing. Juan Pablo Berta del Servicio de Ingeniería de Tránsito de la Intendencia de Montevideo en Agosto del 2014 y con el Ing. Daniel Muniz del departamento de Informática de la Intendencia en Setiembre de 2014 para conocer la situación del trafico capitalino, aprender de su experiencia y obtener datos que nos fueran útiles para el proyecto.

Buscando una aproximación aun mas precisa se realizaron trabajos de campo para determinar la configuración de los semáforos, la densidad de tráfico y el tiempo de recorrida. El mapa y la frecuencia de ómnibus son de acceso publico así como el simulador utilizado.

Se creará un programa que implemente un algoritmo evolutivo multiobjetivo que utiliza un simulador de tráfico para obtener las métricas a optimizar. Se busca obtener una nueva configuración de semáforos que en las simulaciones se comporte mejor que la situación actual basándonos en la velocidad promedio de ómnibus y el resto de los vehículos.

Dada la complejidad del problema el programa sera paralelo y se utilizara la plataforma Cluster fing para poder aumentar el tiempo real de procesamiento. además se intentara luego de estudiado los resultados buscar escenarios alternativos para mejorar los tiempos de viaje en el Corredor.

1.0.4. Limitaciones y alcance

La zona modelada comprende todo el tramo de el Corredor Garzón y dos caminos paralelos a ambos lados que dada la configuración de las calles las cuales corren en diagonal fue un proceso complejo. El revelamiento de datos hecho in-situ fue realizado para un número determinado de calles (se evalúa Garzón y 5 cruces representativos)

para tener una aproximación útil no pretende ser un estudio detallado sobre el trafico en la zona.

Como lo que se pretende modelar es el tráfico vehicular y transporte publico por tanto la simulación de peatones no están incluidos.

1.0.5. Aportes

Este proyecto se encuentra publicado en la pagina web:...

además se desarrollo un informe resumido que fue presentado en...

Cabe destacar que este proyecto se presentó en Ingeniera demuestra 2014, siendo bien recibido por el publico. Constatando de primera mano que la problemática es real y llegando a la conclusión que los Ingenieros tienen las herramientas necesarias para solucionar problemas que afectan directamente a la sociedad.

1.0.6. Estructura del documento

En el capítulo 2 se hace un repaso sobre que es un algoritmo genético y los conceptos relacionados. El capitulo 3 brinda el marco teórico donde se da una introducción al problema del tráfico y el corredor Garzón particularmente, además se presenta el simulador de trafico y otras herramientas utilizadas. En el capitulo 4 se muestran los trabajo relacionados enfocando en algoritmos genéticos para la sincronización de semáforos. En el capítulo 5 se explica la estrategia seguida para la resolución del problema y se da en detalle el diseño de la misma. El capítulo 6 cuenta con tablas, gráficas e información relacionada con el análisis experimental realizado en los distintos escenarios que se eligieron. El capítulo 7 da las conclusiones finales y el trabajo a futuro que se puede realizar.

Capítulo 2

Algoritmos Evolutivos

2.1. Introducción

Uno de los puntos importantes del presente trabajo son los algoritmos genéticos por lo que se dedica esta sección para brindar un repaso por los conceptos y definiciones necesarias para comprender el desarrollo posterior de la solución.

Los algoritmos evolutivos son métodos no determinísticos que se inspiran en la evolución natural de las especies utilizando conceptos como población, cruzamiento, mutación, selección, etc. Estos se utilizan para resolver problemas de optimización y búsqueda, entre otros (Nesmachnow, 2002).

Es una técnica iterativa que busca en cada paso mejorar las soluciones por medio de operadores basado en un criterio predefinido para maximizar o minimizar.

Este tipo de solución ha demostrado su utilidad en una amplia variedad de problemas complejos.

2.1.1. Algoritmos Genéticos

El algoritmo genético es uno de los más populares dentro de los algoritmos evolutivos.

La idea base es que partiendo de una población inicial de individuos se seleccionan los mejores en base a su aptitud respecto a solucionar el problema y estos se utilizan para generar nuevos individuos ya sea por combinación o modificación. Por tanto en cada paso obtenemos mejores soluciones hasta detenernos usando un criterio de parada ya sea el número de iteraciones o cuando ya no se puede mejorar más la solución.

Un individuo es una codificación de la solución que resuelve el problema. La población inicial puede generarse aleatoriamente o basándose en algún conocimiento previo. La función de evaluación indica que tan buena o apta es una solución en comparación con las demás. En cada iteración la cual se llama generación se aplican operadores de cruzamiento estos son formas de combinar a los individuos para obtener otros que potencialmente sean una mejor solución y también cambios aleatorios sobre los individuos llamado mutación.

Por tanto se van seleccionando, combinando y cambiando las mejores soluciones en un proceso que va obteniendo mejores soluciones. El criterio de parada nos indica cuando termina este proceso, ya sea por que se alcanzó un número de generaciones predefinidos o por que la mejora no es evidente. Al final se devuelve la mejor solución encontrada en todo el proceso.

Hay que indicar que no es una técnica exacta pero si logra muy buenas aproximaciones y es muy buena en problemas complejos por su flexibilidad y robustez.

Representación de soluciones

No podemos trabajar directamente sobre las soluciones, por lo que tenemos que codificarlas en un modelo que nos sirva para poder aplicar el algoritmo. La inspiración biológica se ve en los nombres que adopta esta representación, llamada Cromosoma que es un vector de genes y cada valor de un gen se llama alelo. En general se codifica un vector de números binarios o reales de largo fijo, lo que facilita la aplicación de los operadores.

Función de Evaluación

Indica que tan bueno es un individuo para resolver el problema en cuestión con un valor conocido como Fitness. Este se utiliza para seleccionar a los mejores y de esta forma guiar la exploración hacia la mejor solución. Se deben tener en cuenta las restricciones del problema para que las soluciones no factibles no sobrevivan. En general es donde se consume el mayor tiempo del algoritmo en comparación con los demás operadores.

Operador de Selección

Existen diversos operadores de selección , su función es que las mejores características de los individuos se mantengan en las siguientes generaciones. Los tipos más populares son:

- Ruleta: También conocida como selección proporcional elige aleatoriamente individuos en la cual la probabilidad de selección es proporcional al valor de fitness.
- Torneo: Se elige aleatoriamente un determinado número de individuos los cuales compiten entre si.
- Elitismo: Los mejores individuos son mantenidos entre las generaciones.

Cruzamiento

Su función es combinar individuos para lograr mejores soluciones. Existe una tasa que se puede modificar para indicar la probabilidad de que se realice el cruzamiento.

- Cruzamiento de un punto: A partir de dos padres se selecciona un punto al azar de los cromosomas obteniendo dos trozos que se combinan para obtener dos hijos. Se explica en la figura 2.1
- Cruzamiento multipunto: El método anterior se puede generalizar para obtener más puntos de corte y más recombinaciones.

Mutación

Indica el método utilizado para modificar un individuo, esto se realiza para lograr más diversidad y no caer en máximos locales. En general aplica una modificación aleatoria en el cromosoma.También hay una tasa de probabilidad, en general es baja. En el caso de un cromosoma binario se aplica la inversión sobre un alelo.

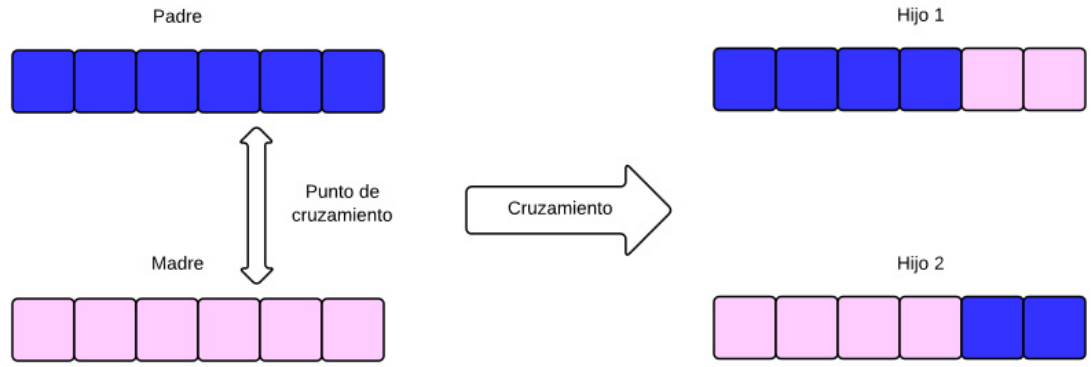

Figura 2.1: Cruzamiento de un punto

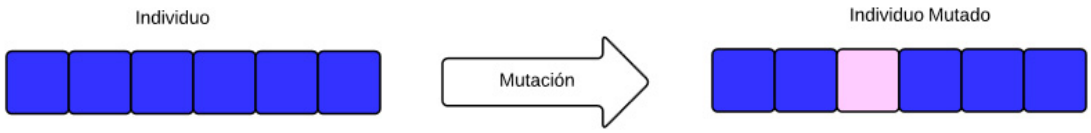

Figura 2.2: Mutacion por inversión binaria

Reemplazo

Se indica cual es el criterio que debemos tomar para generar una nueva población, ya sea tomando solo los hijos creados o comparando también con los padres o aplicando algún otro criterio.

Criterio de parada

Indica cuando debe terminar el algoritmo, puede ser definiendo un número fijo de generaciones o analizando si el mejor valor de fitness se mantiene relativamente constante durante un número determinado de generaciones.

2.1.2. Funcionamiento

El esquema básico de funcionamiento es el siguiente:

2.1.3. Algoritmos genético multiobjetivo

En este caso se busca una solución que satisfaga de forma simultánea tanto las restricciones del problema como varios objetivos distintos.

2.1.4. Algoritmo Genético Paralelo

Los problemas complejos suelen requerir una alta demanda computacional por lo que aplicar técnicas de paralelización es útil para lograr tiempos de ejecución menores.

Algoritmo 1 Algoritmo Genético

```
1: Inicializo( Pob(0))
2: generacion = 0
3: while No llegue al criterio de parada do
4:    Evaluar Pob(generacion)
5:    Padres = Seleccionar(Pob(generacion))
6:    Hijos = Cruzamiento(Padres) y Mutacion(Padres)
7:    NuevaPob = Reemplazar Pob(generacion) con Hijos
8:    generacion++
9: end while
10: return Mejor solución
```

Existen varios niveles de paralelización ya sea a nivel global enfocándonos en paralizar la función fitness, a nivel de la población, o a nivel del individuo. (Nesmachnow, 2002)

En el caso de los algoritmos genéticos gran parte del tiempo se ocupa en la etapa de evaluación, por esta razón es un buen método para distribuir la carga en varios procesadores para que las evaluaciones se realicen en paralelo.

2.1.5. Maestro-Esclavo

En este modelo un proceso maestro es el encargado de realizar los operadores básicos del algoritmo y distribuir a procesos esclavos la evaluación de la función fitness para un conjunto de individuos, el esclavo devuelven el resultado y luego el maestro es el encargado de continuar ejecutando los operadores.

De este modo aumenta la eficiencia computacional del algoritmo ya que una de las funciones más costosas es distribuida entre varios nodos.

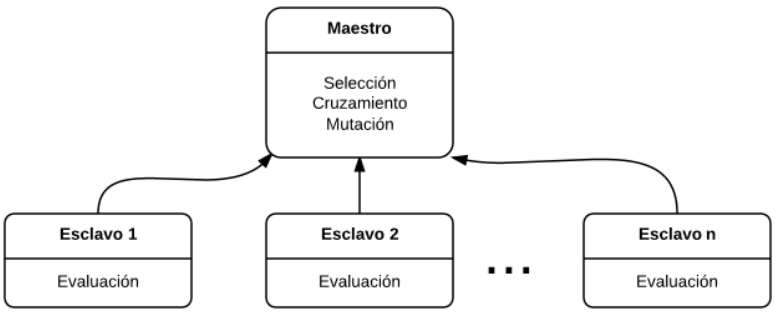

Figura 2.3: Modelo Maestro-Esclavo

Capítulo 3

Marco Teórico

En este capitulo se aborda el marco teórico necesario para comprender más fácilmente el desarrollo de los capítulos posteriores. Se analiza el problema del tráfico en general, los simuladores y la teoría detrás del algoritmo a utilizar.

3.0.6. Problema del tránsito vehicular

En gran parte del mundo se está produciendo un crecimiento sostenido del parque automotor lo que ocasiona una serie de problemas que afectan la calidad de vida de las personas relacionados con el agravamiento de las congestiones vehiculares (CEPAL, 2003).

Este problema tiene un gran impacto en el desarrollo de las ciudades por lo que es un componente principal en los planes estratégicos para su crecimiento.

La congestión ocasiona una progresiva merma en la velocidad promedio de circulación, lo que incrementa la duración de los viajes, aumenta el consumo de combustible y la contaminación atmosférica y sonora, lo que repercute directamente en la salud de las personas. además se genera una exigencia en las vías de tránsito que ocasiona un deterioro mayor de calles y rutas.

Uruguay no escapa a este fenómeno en particular Montevideo, donde el aumento del parque automotor está en ascenso constante desde el 2005 (INE, 2014) Y según proyecciones el crecimiento seguiría en un promedio de 4.5 % anual hasta el 2020. (BBVAResearch, 2013)

Esto viene de la mano con el sostenido aumento de las ventas de vehículos desde el 2003 (Autoanuario, 2014)

Los expertos indican que la congestión ya está instalada y la infraestructura vial no acompasó este crecimiento. además se indica que Montevideo es la ciudad con más semáforos por automóvil en Latinoamerica. Con más de 620 cruces semaforizados, alguno de los cuales no están coordinados.(Subrayado, 2013)

Por todo esto es relevante el tema de la sincronización de semáforos para agilizar el tránsito y no generar congestiones, aumentando la velocidad promedio de los viajes y mejorando las perspectivas de desarrollo de la ciudad así como la calidad de vida de sus habitantes.

3.0.7. Corredor Garzón

El corredor Garzón se ubica en Montevideo Uruguay, fue construido como parte de un plan de movilidad que incluye otros 4 corredores. Presenta un carril exclusivo para ómnibus y preferenciales http://www.montevideo.gub.uy/ciudadania/stm-transporte-metropolitano/plan-de-movilidad/corredores

Tiene 6km de largo , extendiéndose desde ... hasta ..

Agregar mapa y poner referencia de donde se saco

Los problemas de sincronización de semáforos fueron admitidos en varias publicaciones.

18 diciembre 2013 - Corredor Garzón lucha contra el tiempo Dice que antes de Garzon se demoraba promedio 18 minutos, y al inaugurar el corredor 30 minutos. Después se mejoro algo para equilibrar los tiempos Para el jerarca, eso se dio çon la diferencia de que hoy hay 15 semáforos más y se ganó en seguridad". En concreto, tras la obra, se pasó de tener 5 semáforos a 20. Según Campal, su des-coordinación inicial, entre otros aspectos, fue lo que provocó tales demoras, generando malestar en los usuarios. inversión de 60 millones

4 agosto 2013 - Garzon desde un omnibus

30 julio 2013 - Intendetnta admite errores La intendenta admte errores y dice: no se ha logrado sincronizar los semáforos. Hay un tema con el software,(y) la empresa subcontratada no ha dado los resultados esperados

Abril 2013 - Otro Corredor con obras paralizadas por criticas a GArzon

3.1. Simulación y herramientas

3.1.1. Simuladores de tráfico

Los simuladores de tráfico son programas que simulan el movimiento de vehículos sobre una red de calles, es una herramienta muy usada en la investigación de tráfico vehicular; así como estudio de congestiones o análisis de impacto que tendrán nuevas infraestructuras. Las razones para usar una simulación son varias, entre ellas se encuentra la rapidez, ya que la simulación se puede realizar en tiempo mucho más rápidos que en la realidad, el costo en dinero pues no estamos afectando el escenario real y tampoco tenemos que modificar o detener el escenario real para probar nuevos parámetros. además nos sirve para poder prever situaciones que podrían darse bajo determinadas circunstancias.

Los simuladores se pueden dividir en microscópicos o macroscópicos según el nivel de detalle de la simulación. Un simulador macroscópico modela el tráfico vehicular como un fluido. En cambio un simulador microscópico simula el movimiento de cada vehículo según sus características particulares.

SUMO(SUMO, a) es uno de los simuladores abiertos más populares, es microscópico y utiliza una serie de archivos XML que representan las rutas, los vehículos y el tráfico.

Cuanto más crece el número de vehículos y la complejidad de la red de mapas más difícil se hace crear la entrada básica que necesita el simulador. Aunque existen diversas herramientas que ayudan a este proceso aún se requiere un trabajo manual para el acondicionado de estos archivos.

En las siguientes secciones se muestra más en detalle algunas de sus funcionalidades y características.

3.1.2. Modelo de tráfico

Esta es la representación de la circulación de vehículos, aquí exponemos algunos de los más populares.

Aleatorios: Genera diferentes recorridos que seguirán los vehículos aleatoriamente

JTR (junction turning ratio) : basados en probabilidades en los cruces es decir cuando un vehículo llega a un cruce tiene cierta probabilidad de seguir o doblar.

Basado en áreas: Se especifican áreas como un conjunto de calles y se realizan recorridos entre ellas.

Basado en Actividad: Se especifica la cantidad de casas, vehículos y población en un determinado lugar y el modelo genera la densidad de tráfico que se producirá basado en los tipos de actividades que se realizan comúnmente como ir al trabajo, hacer las compras, ir a la escuela, etc

Definido por el usuario: que permiten determinar la ruta de los vehículos con mayor detalle.

3.1.3. Representaciones

Red de calles

La red de calles se representa como un grafo dirigido en un archivo xml con extensión .net.xml . Allí se especifican los nodos, y vértices así como sus atributos. También se representan los semáforos. Esto se genera utilizando una herramienta para convertir un mapa al formato que SUMO utiliza.

Representación tráfico

En este caso también se utiliza un archivo xml donde se definen las rutas y los recorridos. Un recorrido representa el movimiento de un vehículo de un punto inicial hacia un punto final (El recorrido se hacen en tiempo de ejecución utilizando el camino más corto basado en el tráfico).

Representación del tiempo

El tiempo se representa como una serie de pasos discretos, cada uno durando un segundo. Este valor se puede modificar aunque es recomendado dejarlo así para que sea consistente.

Tipos de vehículos

Se pueden crear diferentes tipos de vehículos especificando propiedades como largo, velocidad máxima, aceleración, color, etc. También cuenta con vehículos por defecto como buses, camiones o automóviles.

3.1.4. Configuración de la simulación

En esta sección se realiza un resumen sobre los pasos realizados para tener los elementos necesarios para realizar la simulación.

Diseño del mapa

El mapa base de la zona proviene de Open Street Map (OSM, a), luego se cotejó su exactitud con Google Maps y Bing Maps.

Figura 3.1: Tramo del mapa de Garzon de OSM

Se utilizó la herramienta netconvert para convertirlo al formato que SUMO acepta. Para esto se realizaron varios ajustes editando los archivos xml para corregir errores en las calles, cruces y conexiones para que fuese fiel a la realidad y mantuviera la compatibilidad con SUMO.

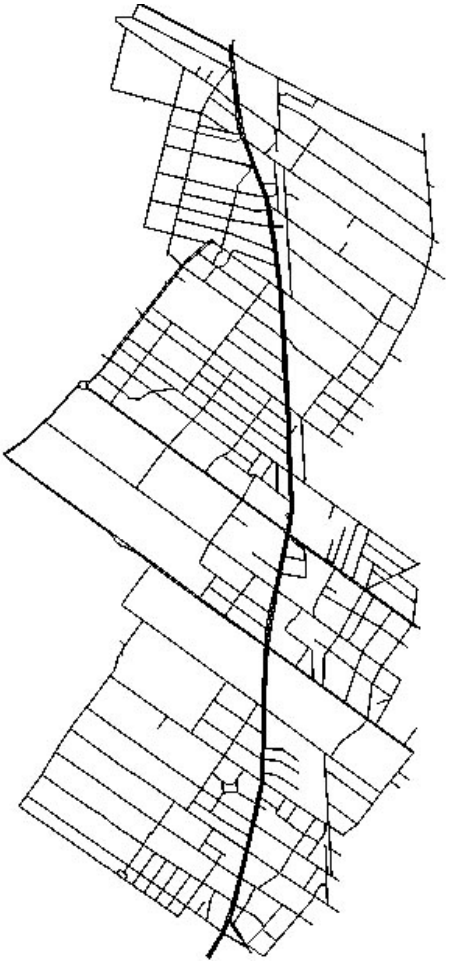

Figura 3.2: Mapa cargado en SUMO luego del modelado y procesamiento

Trabajo de campo realizado

Al no contar con datos públicos sobre la configuración de los semáforos de la zona, realizamos un revelamiento in-situ con las siguientes características.

Se seleccionaron 5 cruces que presentan diferentes características para poder modelar las variantes tanto de cruces con alto tráfico, medio y bajo. Estos son: Camino Ariel, Battle y Ordoñez, Plaza Videla, Camino Colman y Aparicio Saravia.

Se eligió el día miércoles, que estuviera soleado y entre las 15 y 17 horas para no tener los sesgos que se producen los fines de semana o días de lluvia. Se realizaron filmaciones de 30 min en los cruces y luego se analizaron los vídeos para realizar el conteo manual con la posibilidad de enlentecer el vídeo para mayor facilidad. Luego se completa una planilla excel donde se tiene la información de vehículos que recorren Garzón, la calle que cruza y los que doblan. También discriminamos entre vehículos y ómnibus que recorren Garzón.

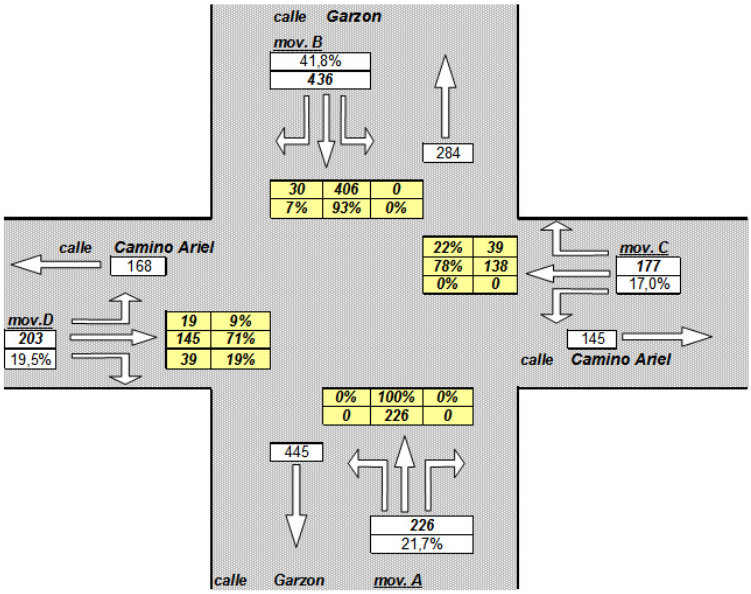

Figura 3.3: Planilla electrónica para el conteo manual en Camino Ariel

además se realizaron recorridas de punta a punta del corredor a una velocidad constante para verificar los tiempos obtenidos en la simulación para este recorrido.

Para obtener la configuración de los semáforos se realizo un recorrido en bicicleta por el corredor cronometrando la duración del tiempo en cada esquina de cada semáforo. Tanto de ida como de vuelta para corroborar que fueran correctos. Estos tiempos fueron verificados por los vídeos obtenidos.

Creación del modelo del tráfico

Con los datos relevados se creo un modelo vehicular de la ciudad [poner el mapa] que brinda una aproximación sobre la densidad de tráfico y la velocidad promedio de circulación.

Se utilizó el programa Traffic Modeler (Papaleontiou) con lo que se logra generar modelos de tráfico complejo de manera visual y sencilla. Se optó por un modelo de movilidad entre áreas lo que permite una buena granularidad al especificar la densidad de tráfico.

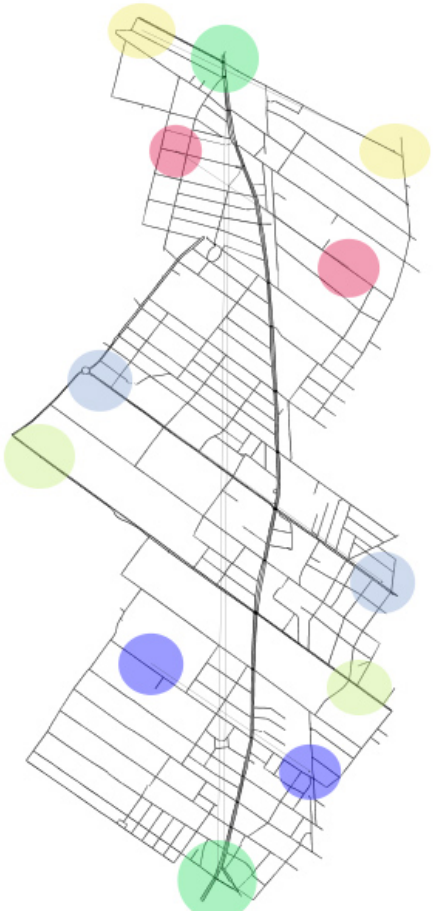

Figura 3.4: Mapa del TrafficModeler con las áreas de tráfico. Círculos del mismo color indican tráfico especificado entre esas áreas

Actualmente no se cuenta con información publica relativa a la configuración de los semáforos en la ciudad, aunque cabe destacar que en el futuro se prevé crear un sistema centralizado de sincronización de semáforos. (Observador) En este caso se agregó la información sobre la configuración de los semáforos de los datos relevados.

Se manejaron dos tipos de vehículos autos y ómnibus cada uno con características distintas tanto de tamaño, aceleración y velocidad máxima.

Se agregaron las frecuencias y los distintos recorridos de los ómnibus obtenidos de datos públicos de la Intendencia de Montevideo (IMM) Estos incluyen las lineas urbanas 'G', la '409' , '522' y '148' . Las líneas de ómnibus suburbanas realizan un mismo trayecto

y las generalizamos con el nombre 'A'. Todas las líneas de ómnibus fueron cargadas con las paradas correspondientes y se hicieron variantes en los viajes dentro de una misma línea para simular el hecho de que no siempre se para en las mismas paradas.

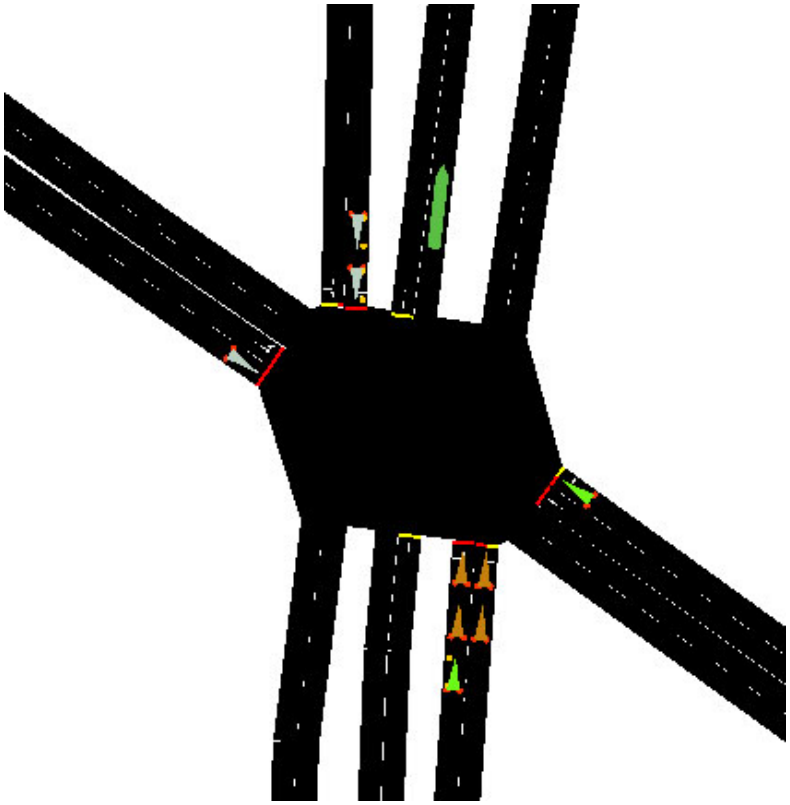

Figura 3.5: Simulacion de tráfico en SUMO en el cruce entre Bulevar Battle y Ordoñez y el Corredor Garzon.

3.2. Herramientas

Aquí se muestra información útil sobre las herramientas utilizadas.

3.2.1. Open Street Map:

Es un proyecto en donde una comunidad crea mapas libres y editables (OSM, a). Cuenta con cerca de 1.8 millones de usuarios y más de 20.000 que editaron algo en el ultimo mes (OSM, b) por lo que es muy activa. Se utilizan datos de GPS móviles, fotografías satelitales y otras fuentes libres para crear los mapas y editarlos.

3.2.2. SUMO (Simulation of Urban MObility)

Es un simulador de tráfico gratis y abierto que nos permite modelar redes de calles, vehículos, transporte publico, ciudadanos y semáforos (SUMO, a). Sigue un modelo microscópico ya que realiza la simulación individual explicita de cada elemento. además incluye un conjunto de herramientas destinadas a facilitar la generación de tráfico, construcción de mapas, etc.

3.2.3. Por que usar SUMO?

otras alternativas? Revisar estado del arte? Sumo es gratis Nos permite utilizarlo sin interfaz gráfica por linea de comando lo que aumenta sensiblemente la velocidad ed ejecución, y nos permite visualizar la interfaz gráfica una vez completada la optimización para ver exactamente como se comporta el sistema en la simulación.

Sumo presenta varias salidas con información interesante: (SUMO, b) La salida obtenemos el tiempo de simulación, la cantidad de emitidos y la cantidad de completados. Buscamos que sea ¿80Permite la ejecución tanto por linea de comando como una interfaz gráfica para visualizar

3.2.4. NetConvert

Utilizado para la generación de la red a partir de un mapa. Por ejemplo podemos transformar un mapa de Open Street Map en archivo XML de red que SUMO reconoce. Este programa viene integrado con SUMO

3.2.5. DUaRouter

Genera recorridos de vehículos basado en dinámicas definidas por el usuario. Esta utilidad viene integrada con SUMO.

3.2.6. Traffic Modeler

Herramienta para la generación de tráfico de manera visual y luego transformarlo para que sea reconocido por SUMO. (Papaleontiou)

Escenarios

Capítulo 4

Trabajos relacionados

4.1. Introducción

La investigación del estado del arte se realizo con dos objetivos en mente, el primero analizar las distintas soluciones que existen actualmente para el problema y segundo encontrar nuevas prácticas, algoritmos o utilidades que pudieran fortalecer la solución.

El problema del tráfico optimizando las luces de los semáforos se puede resolver por muchos métodos como redes neuronales (López et al., 1999), lógica difusa (G. Y. Lim and Hong, 2001), redes de petri (DiFebbraro et al., 2002), etc; por lo tanto la cantidad de soluciones encontradas fue abundante y variada por esto se decidió enfocarse en soluciones lo más cercanas a la propuesta y en otras que tuvieran alguna particularidad interesante para destacar.

- J. Sánchez, M. Galán, and E. Rubio. Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization. 2004

  Este trabajo se basa en tres puntos: El uso de algoritmos genéticos para la optimización , simulación de autómatas celulares para la función de evaluación del tráfico, y un cluster para realizar ejecuciones en paralelo. El modelo es pequeño con 5 calles de 2 vías que se intersectan. La codificación del cromosoma es una tira de números enteros, donde se codifica para cada intersección cual calle está habilitada en cada ciclo. Usa una estrategia de selección elitista donde los 2 mejores se clonan a la siguiente generación, y el resto es generado por cruzamiento de 2 puntos.

  Para la evaluación se usa el tiempo medio, esto es desde el momento que un vehículo entra en la red hasta que sale. Se utilizo un cluster y programación paralela utilizando MPI2[] con una estrategia master-slave, el master envía los cromosomas a los esclavos que evalúan y devuelven el resultado y luego el master se encarga de generar la siguiente población.

  Se compararon los resultados con una simulación aleatoria y con una fija, obteniendo la solución propuesta mejores resultados en todos los casos evaluados

  Este mismo grupo realizo trabajos similares expandiendo esta investigación, como los que siguientes.

- J. Sánchez, M. Galán, and E. Rubio. Applying a traffic lights evolutionary optimization technique to a real case: "las ramblas" area in santa cruz de tenerife. 2008
  Lo interesante de este estudio es que se aplica lo expuesto en el trabajo anterior en

un lugar real (Santa Cruz de Tenerife) para validar los resultados. Algunas mejoras que se introdujeron fueron que el cromosoma se codifica utilizando código Gray lo que dicen mejora el rendimiento en mutación y cruzamiento. La población inicial son nueve "soluciones" provistas por la alcaldía de la ciudad. Tanto la estrategia de selección como de cruzamiento y mutación es similar al anterior trabajo.

El modelo se discretizó quedando en 42 semáforos, 26 entradas y 20 salidas. Las soluciones provistas por la alcaldía se simularon y se utilizo para comparar con los resultados obtenidos por el algoritmo que en términos generales logra un aumento del rendimiento de hasta 26 %.

- J. Sánchez, M. Galán, and E. Rubio. Traffic signal optimization in "la almozara" district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing. 2010 Este trabajo es similar al anterior pero se destacan algunos cambios, por ejemplo se probaron 4 diferentes funciones de fitness: Cantidad de vehículos que llegaron a destino, tiempo de viaje promedio, tiempo de ocupación promedio, velocidad promedio global. También agrega medidas correspondientes al gas total emitido por los vehículos que tiene relación con la velocidad a la que van. El modelo discretizado de la zona de "La Almozara" cuenta con 17 semáforos, 7 intersecciones, 16 entradas y 18 salidas. Se simuló tanto un caso estándar como casos de alta congestión de tráfico, las comparaciones se hacen respecto a las distintas funciones de fitness y los distintos escenarios planteados logrando buenos resultados.

- J. Penner, R. Hoar, and C. Jacob. Swarm-based traffic simulation with evolutionary traffic light adaptation. 2002 Este trabajo se centra en un modelo de simulación basado en enjambres que luego se optimiza utilizando un algoritmo genético cuya función de fitness es el tiempo promedio de los vehículos dentro de la red. El cromosoma cuenta con la secuencia y duración de los semáforos, así como la relación con los semáforos complementarios, la mutación tiene en cuenta esto para que no ocurra en una misma intersección dos luces verdes. El cruzamiento se hace entre los distintos semáforos con una probabilidad más alta si esta en la misma intersección.

  El modelo cuenta con una ruta de 2 vías, con 3 carriles, y 3 intersecciones con 1 ruta de 2 vías y un solo carril. Se comparan 3 escenarios distintos obteniendo mejoras significativas con respecto al inicio.

  Luego se realiza otro escenario más complejo de 28 semáforos y 9 intersecciones logrando buenos rendimientos de hasta 26

- D. H. Stolfi. Optimizacion del trafico rodado en ciudades inteligentes. 2012 Este trabajo se basa en el concepto de una ciudad inteligente enfocando en la movilidad ya que indica que los atascos del tráfico provocan no solo perdidas económicas sino también contaminación ambiental.

  Para ello utiliza un algoritmo inteligente que tomando en cuenta el estado de congestión de las rutas sugiere al usuario cual es la ruta más rápida a su destino, utilizando un dispositivo en el automóvil que se enlazara por wifi con los semáforos que cuentan con sensores. Por lo tanto el trabajo no se basa en la optimización de las señales de los semáforos existentes sino agrega encima de esto un sistema de búsqueda de mejor ruta.

Para el modelo utiliza una zona de la ciudad de Málaga obtenido desde (OSM, a), cuenta con 8 entradas y 8 salidas, para la simulación utiliza (SUMO, a). Los vehículos modelados son: turismo, monovolumen, furgoneta, camión donde se varia la longitud, velocidad y probabilidad que entre en la red de tráfico.

Se intenta minimizar los tiempo de viaje de los vehículos que circulan por la red. Para ellos se utiliza un algoritmo genético cuya estrategia de selección consiste en tomar los 2 peores individuos y reemplazándolo por los 2 mejores hijos encontrados. En el cromosoma se representa cada sensor, con los destinos y rutas posibles. La función de fitness tiene en cuenta la cantidad de viajes completados durante el tiempo de ejecución, el tiempo medio utilizado, y el retraso medio. Se prueban varias estrategias de cruzamiento y mutación. Las ejecuciones tienen un tiempo fijo de duración.

Compara el resultado con una simulación donde se generaron 64 itinerarios diferentes, esto se prueba en 3 escenarios diferentes. Las simulaciones se realiza hasta con 800 vehículos,se concluye que al aumentar la cantidad de vehículos (más de 400) en el sistema la solución mejora sustancialmente el resultado base.

- K. T. K. Teo, W. Y. Kow, and Y. K. Chin. Optimization of traffic flow within an urban traffic light intersection with genetic algorithm. 2010 Este trabajo presenta un modelo simple con una sola intersección en donde se intenta optimizar los tiempos de los semáforos para lograr mejor rendimiento. El cromosoma representa los tiempos de la luces verdes mientras que la función de fitness es el largo de las colas generadas. Un aspecto interesante es que la simulación tiene un tiempo fijo de 600 segundos por generación pero no se detalla el tipo que se utilizo. Las conclusiones indican que la optimización usando algoritmos genéticos es buena para el problema del flujo de tráfico.

- D. J. Montana and S. Czerwinski. Evolving control laws for a network of traffic signals. 1996 Esta propuesta utiliza un enfoque adaptativo con sensores que analizan el tráfico en tiempo real (un sensor para saber cuantos autos pasan y otro para saber que tan larga es la cola) tomando en consideración los cambios que se producen con respecto al caso promedio y cambiando los tiempos de las señales en forma acorde. La premisa se basa en la inteligencia colectiva en donde agentes individuales realizan tareas simples que al interactuar producen resultados globales.

  Se aplica programación genética más específicamente STGP (strongly typed genetic programming) (Montana, 1995) que aprende el árbol de decisión que sera ejecutado por todas las intersecciones cuando decida el cambio de fase. Además un algoritmo genético híbrido busca diferentes constantes que serán usadas en los arboles de decisión mejorando el flujo de tráfico.

  La medida básica de efectividad en la función de evaluación es el “Delay”, esto es el total de tiempo perdido por causa de las señales de tráfico. Se probaron 3 modelos distintos que tienen 4 intersecciones con una versión especial del simulador TRAF-NETSIM (Rathi, 1990)

  El experimento arroja buenos resultados en cuando a la performance de la red y destaca la buena adaptabilidad en diferentes circunstancias. Aunque se marca el hecho de que el modelo es simple y de tamaño pequeño, siendo una incógnita como funcionara con problemas más complejos.

- A. Vogel, C. Goerick, and W. von Seelen. Evolutionary algorithms for optimizing traffic signal operation. 2000

  La solución utiliza un enfoque auto-adaptable para mejorar el tráfico tanto en el corto como el largo plazo a través de la optimización de las señales de tráfico en las intersecciones de una red de rutas. Al darle dinamismo a cada intersección se mejora el rendimiento de la red.

  Destaca el hecho que dada una configuración de semáforos aún siendo optimizada usando simulaciones es difícil que sea la mejor en todas las situaciones o en casos extremos (horas picos). Para solucionar esto proponen un sistema auto-adaptable que toma la información del tráfico actual usando detectores de vehículos y espacios.

  Utiliza el concepto de fases para representar las distintas posibilidades en la señalización de la intersección, y cuanto tiempo debe permanecer en esa fase. Esto provoca que cuanto más fases más cantidad de secuencias son agregadas. Propone el desarrollo de un algoritmo evolutivo donde cada individuo representa un sistema de fases mientras el fitness se obtiene simulando ese sistema en un modelo de tráfico. Este modelo es relativamente pequeño con una intersección con 4 brazos, cada uno con 3 lineas donde la ruta principal tiene el doble de densidad vehicular. El simulador utilizado está basado en SIMVAS++.

  Los resultados indican que la ventaja de usar conocimiento experto para configurar los parámetros iniciales es mínimo ya que llega muy rápido a resultados similares. Tanto la búsqueda de los mejores parámetros como en estructuras más simples el algoritmo se comporta con buenos resultados.

- N. M. Rouphail, B. B. Park, and J. Sacks. Direct signal timing optimization: Strategy development and results. 2000

  Se estudia una pequeña red de tráfico de 9 intersecciones con semáforos en la ciudad de Chicago(Us), contando con trafico de vehículos, parking, rutas de ómnibus y paradas. Se toman valores reales en horas pico, comprobando que las colas que se generan en la simulación coinciden con la realidad. Usa el programa (TRA) que permite visualizar mapas y contiene optimización de varios algoritmos genéticos y (COR) un simulador de tráfico comercial. Se probaron 12 estrategias distintas de resolución distintas midiendo el tiempo de demora en la red y el largo de las colas producidas. Los resultados indican que la performance de la red aumentó considerablemente usando este método.

4.2. Resumen

Aquí un breve repaso sobre los trabajos evaluados y su comparación con la propuesta presentada.

El trabajo de (Sánchez et al., 2004) posee algunos puntos de contacto como es la ejecución paralela en un cluster y la arquitectura master-slave. La principal diferencia es que el escenario que evalúan es muy pequeño en comparación y no se compara con un escenario real.

El siguiente trabajo de (Sánchez et al., 2008) expande lo anterior y lo utiliza en un caso real en Santa Cruz de Tenerife siendo de un porte similar a Garzón en términos de

cantidad de cruces y semáforos. Los resultados obtenidos son muy positivos obteniendo mejoras de hasta 26

Se destaca de (Sánchez et al., 2010) donde se prueban diferentes funciones de fitness teniendo en cuenta diversos factores como tiempo de viaje o velocidad promedio. Este trabajo inspiro la realización de una función multiobjetivo que tuviera en cuenta la velocidad promedio en el proyecto actual.

Aunque (**?**) no optimiza la configuración de los semáforos si plantea una posibilidad interesante para mejorar el trafico en una ciudad indicando a los vehículos la mejor ruta por lo que se podría tomar como un elemento en trabajos futuros.

Tanto los trabajos de (Teo et al., 2010) como (Stolfi, 2012) plantean la simulación con un tiempo fijo lo que se utilizó en el proyecto.

Tanto (Montana and Czerwinski, 1996) como (Vogel et al., 2000) proponen algoritmos que se adapten en tiempo real por lo que se destacan como posibles trabajos a futuro. Es digno de mención que todos los trabajos destacan una mejora en el rendimiento al utilizar algoritmos genéticos.

En conclusión el estudio de los trabajos relacionados permitió conocer mas en profundidad distintas soluciones y métodos que fueron tenidos en cuenta en menor o mayor medida en la solución propuesta. El hecho de que se obtuvieran buenos resultados motivo aun mas el desarrollo del trabajo presentado.

Capítulo 5

Estrategia de resolución

En este capitulo se presenta el algoritmo desarrollado, su estructura, parámetros y configuración.

Se comienza explicando la librería Malva utilizada para el desarrollo del algoritmo y la arquitectura especifica que se realizo para resolver el problema.

5.1. Por que usar algoritmo genéticos?

El problema de sincronización de semáforos es NP-Hard y no existe (hasta el momento) un método determinístico que lo resuelva, se buscará mediante algoritmos evolutivos llegar a una configuración aceptable minimizando los tiempos de espera de los automóviles, mejorando así la configuración actual de los semáforos del corredor de Garzón.

5.2. Arquitectura de la solución

En el siguiente diagrama muestra la arquitectura propuesta para el problema.

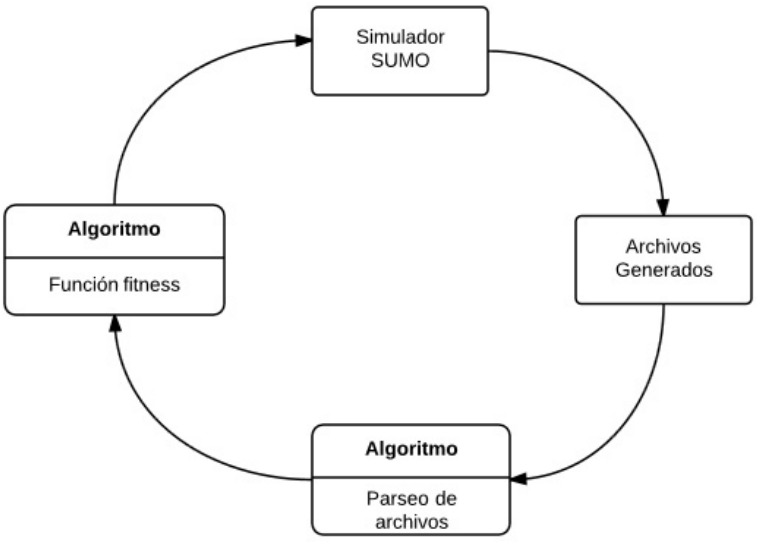

Figura 5.1: Arquitectura del algoritmo

Malva se utiliza para la implementación del algoritmo, en cada evaluación de la función de fitness se realiza un llamado al simulador SUMO.

El simulador genera archivos con información sobre la simulación que son usados por el algoritmo para determinar el fitness.

Luego los archivos son procesados por el algoritmo para extraer los datos útiles necesario para la función de fitness.

5.3. Librería Malva

(Malva) surge como una variante del proyecto (Mallba). Propone la actualización, mejora y desarrollarlo como un proyecto de código abierto colaborativo. Su objetivo es proveer de varios esqueletos de heurísticas de optimización que puedan ser utilizados y extendidos de manera fácil y eficiente.

Los esqueletos se basan en separar dos conceptos: El problema concreto que se quiere resolver y por otro lado el método utilizado para resolverlo. Por tanto un esqueleto se puede ver como una plantilla genérica que se instancia para resolver un problema particular, manteniendo todas las funcionalidades genéricas.

Utiliza el lenguaje c++ dado su alto nivel, modularidad y flexibilidad. Los esqueletos se ofrecen como un conjunto de clases requeridas que son las que el usuario deberá modificar para adaptarlo a su problema y las provistas. que incluyen todos los aspectos internos del esqueleto y on independientes del problema particular. Entre los algoritmos provistos se encuentra el de Algoritmos genéticos y (CHC).

5.4. Especificación del Algoritmo Genético utilizado

Se utiliza el algoritmo genético proviso por la librería Malva llamado NewGA al cual se le realizan algunas modificaciones para que se ejecute en paralelo.

Resumen de las características:

- Algoritmo paralelo: Utiliza el método maestro esclavo para que en cada iteración el maestro genere un hilo para cada ejecución de la función fitness y luego espere a la terminación de todos los hilos para consolidar los datos.
- Función Multiobjetivo: Se intenta optimizar tanto la velocidad promedio de vehículos como de ómnibus teniendo cada uno un peso especifico.
- Representación del cromosoma: Es un vector de números reales que representan los tiempos de los semáforos.
- Cruzamiento y mutación: Se utiliza una variante del cruzamiento por un punto especifico para nuestro caso así como para la mutación.
- Selección y reemplazo: Reemplaza padres por hijos, la selección de los padres se realiza por el método de torneo de 3 individuos, y la selección de hijos por el método de ruleta.

más adelante se realizara el ajuste paramétrico para determinar tanto el tiempo de simulación, criterio de parada y tasas de cruzamiento y mutación optimas para el algoritmo.

El siguiente esquema muestra el algoritmo utilizado. (MalvaAlgoritmo)

Algoritmo 2 Algoritmo Genético de Malva

```
1: t = 0
2: Inicializo( P(t))
3: Evaluar estructuras en ( P(t))
4: while No termine do
5:    t++
6:    Seleccionar C(t) de P(t-1)
7:    Recombinar estructuras en C(t) formando C'(t)
8:    Mutar estructuras en C'(t) formando C"(t)
9:    Evaluar estructuras en C"(t) generando un hilo de ejecucion por cada una
10:   Consolidar valores de la evaluacion
11:   Reemplazar P(t) de C"(t) y P(t-1)
12: end while
```

5.4.1. Representación del cromosoma

Se explican algunas definiciones necesarias para comprender la representación.

Cruce: Es el lugar de intersección de dos o más vías de circulación. Fase: Es la configuración de las luces de los semáforos en un determinado cruce.

El cromosoma se va a agrupar lógicamente en cruces siendo el valor de cada gen el tiempo que demora una fase de un cruce, además se agrega para cada cruce un número

que representa en que fase comienza el semáforo. Por tanto el tamaño del cromosoma depende tanto de la cantidad de cruces como la cantidad de fases de cada uno.

En la representación se omiten las luces amarillas ya que no modifican los tiempos reales del paso de vehículos.

Es importante que el algoritmo no genere soluciones inviables por lo que no debe modificar la combinación de luces de cada fase evitando combinaciones de luces erróneas. Por tanto tanto la modificación se realizara solo en el valor que indica el comienzo de fase y en los tiempos relacionados.

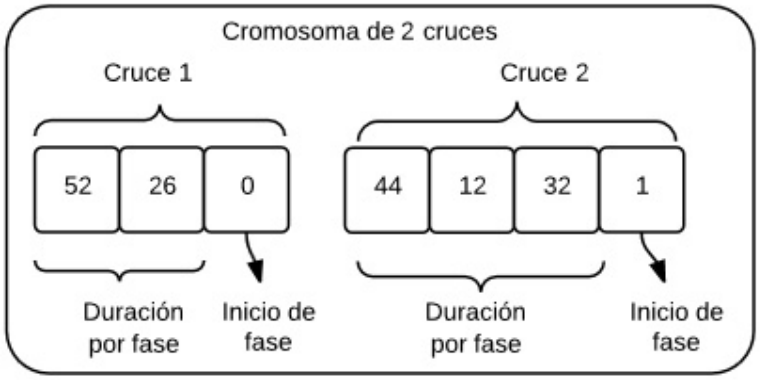

Figura 5.2: Cromosoma de 2 cruces

En la siguiente figura vemos la representación de los archivos de simulación que nos provee SUMO para el cromosoma anterior, donde vemos como se representan las fases por ejemplo el texto “GGGGrrGGGGrr” demora 52 segundos. “G” es Verde, “r” es Rojo e “y” es Amarillo. El offset indica el inicio de la fase.

```
<add>
    <tlLogic id="cluster_-46_-48_-50" type="static" offset="0">
        <phase duration="52" state="GGGGrrGGGGrr" />
        <phase duration="3" state="yyyyrryyyyrr" />
        <phase duration="26" state="rrrrGGrrrrGG" />
        <phase duration="3" state="rrrryyrrrryy" />
    </tlLogic>
    <tlLogic id="cluster_1858033111_917381626_917457905" type="static" offset="1">
        <phase duration="44" state="rrrrrrrGGGGrrr" />
        <phase duration="3" state="rrrrrrryyyyrrr" />
        <phase duration="12" state="rrrrGrrrrrrGrr" />
        <phase duration="3" state="rrrryrrrrrryrr" />
        <phase duration="32" state="GGGGrrrrrrrrrr" />
        <phase duration="3" state="yyyyrrrrrrrrrr" />
    </tlLogic>
</add>
```

Figura 5.3: Representación de Sumo

5.4.2. Inicialización

Para la inicialización de la población se toma como referencia la configuración obtenida con los datos in-situ, luego para cada cruce se hacen variar las duraciones de las fases

de manera aleatoria entre un rango de 5 a 60 segundos (valores configurables) además la fase inicial se elige aleatoriamente entre la cantidad de fases del cruce (se cuentan las luces amarillas).

5.4.3. Función fitness

La evaluación de un individuo se realiza generando un archivo con la configuración de los semáforos en base a su cromosoma y ejecutando el simulador SUMO utilizando esta configuración para luego obtener los tiempos necesarios para calcular el fitness. Al ser una función multiobjetivo los datos que extraemos son la velocidad promedio de los ómnibus y la velocidad promedio del resto de los vehículos.

Esta es la formula de fitness donde x e y indica el peso que le podemos especificar a la función.

f = x.vb + y vv .

5.4.4. Operador de Cruzamiento

Se utilizará cruzamiento de un punto, implementado específicamente para el problema, seleccionando el intervalo entre 2 cruces como punto de corte, por tanto si un tramo del corredor tiene un buen comportamiento esta propiedad lo mantendrá.

Operador de Mutación

La mutación también fue implementada específicamente para el problema, utilizaremos dos tipos de mutación:

- Mutación de duración de fase: para cada fase de cada cruce se hace variar su duración sumando o restando una cantidad dada de segundos entre un rango determinado con una probabilidad dada.
- Mutación de inicio de cruce: se elige aleatoriamente una fase con la cual va a arrancar inicialmente el cruce con una probabilidad dada.

5.5. Resumen

Capítulo 6

Análisis Experimental

En esta sección se presenta la descripción de los escenarios , la plataforma de ejecución y el análisis experimental.

Este se divide en dos etapas, primero se realiza la configuración paramétrica para encontrar la mejor ejecución del algoritmo. Luego se realizan las pruebas donde se comparan los resultados entre los distintos escenarios.

6.1. Desarrollo y Plataforma de ejecución

Los algoritmos fueron desarrollados usando la librería Malva que fue extendida en el código base para soportar la creación de nuevos hilos de ejecución para lograr el funcionamiento en paralelo.

Los escenarios fueron ejecutados en el cluster fing.

Cluster: Es un conjunto de computadoras independientes conectadas para que trabajen integradas como un solo sistema. De esta forma se consigue un alto rendimiento en la ejecución de tareas.

Cluster Fing: Es una infraestructura de alto desempeño, que brinda soporte en la resolución de problemas complejos que demandan un gran poder de computo.

Descripcion del hardware:

- 9 servidores de cómputo

  Quad core Xeon E5430, 2x6 MB caché, 2.66GHz, 1.333 MHz FSB.

  8 GB de memoria por nodo.

  Adaptador de red dual (2 puertos Gigabit Ethernet).

  Arquitectura de 64 bits.

  Servidor de archivos: 2 discos de 1 TB, capacidad ampliable a 10 TB.

  Nodos de cómputo: discos de 80 GB.

- Switch de comunicaciones

  Dell Power Connect, 24 puertos Gigabit Ethernet.

- Switch KVM (16 puertos) y consola.

- UPS APC Smart RT 8000VA.

6.2. Ajuste de parámetros de algoritmos

Se busca la mejor configuración inicial de los parámetros realizando pruebas experimentales con diferentes combinaciones.

- Tiempo de simulación
- Criterio de parada
- Tamaño de la población
- Probabilidad de mutación
- Probabilidad de cruzamiento

Para la realización de las pruebas se generan tres instancias con densidades de tráfico diferentes para realizar las pruebas de configuración. De esta forma el algoritmo queda generico y no sesgado a un caso en particular. Se representa la cantidad de ómnibus y otros vehículos que circulan por la zona de Garzon en un tiempo de una hora. El caso de tráfico medio representa una aproximación de los datos obtenidos in-situ.

- Tráfico Bajo: 30 ómnibus y 500 vehículos
- Tráfico Medio: 60 ómnibus y 1000 vehículos
- Tráfico Alto: 120 ómnibus y 2000 vehículos

En general se realizan 21 pruebas individuales en cada prueba para lograr mejor confiabilidad estadística, 7 para cada una de las densidades de tráfico.

En este proceso primero se define el tiempo de simulación y el criterio de parada, luego de establecidos se realizan las pruebas para todas las combinaciones de tasa de cruzamiento y mutación buscando los mejores valores para optimizar el algoritmo.

Al estas utilizando un cluster tenemos a nuestra disposición tanto la métrica del tiempo real que llevo la ejecución, así como también el tiempo secuencial es decir la suma del tiempo de procesamiento de todos los procesadores involucrados en la evaluación del algoritmo. A saber en las comparaciones del tiempo de ejecución se refiere al tiempo secuencial que es más confiable al no tener el sesgo dado por la cantidad de procesadores utilizados en la evaluación. En general para las diversas pruebas se utilizaron entre 16 y 64 procesadores.

6.2.1. Tiempo de simulación

Para tener un mejor control sobre los tiempos totales de ejecución, se busca encontrar un número fijo para el tiempo de la simulación de cada escenario que se ejecutará para cada solución.

Teniendo en cuenta que cada simulación de los escenarios representan el tráfico vehicular durante una hora real y que se validó que en cada escenario más de los 80 % de los vehículos hayan dejado la simulación, es decir llegado a sus destinos.

Se establece un tiempo de simulación de 4000 steps (medida interna de tiempo del simulador SUMO equivalente a 66 minutos reales) que cumple con los criterios establecidos.

6.2.2. Criterio de Parada

Se elige como criterio de parada el número de generación, esto permite estandarizar las pruebas para una mejor comparación.

Para determinar el número de generación donde parar el algoritmo se busca un compromiso entre un buen resultado y un tiempo de ejecución apropiado que no sea excesivo.

Para esto se decide que por un lado la ejecución del algoritmo deberá estar comprendida entre 1h y 24h y además comprobar experimentalmente que el valor de fitness no tiene una gran variación en las ultimas 100 generaciones.

Luego de la realización de las pruebas se elige el número de 500 generaciones como criterio de parada, como se ve en la siguiente gráfica representativa de las pruebas realizadas se aprecia como el valor de fitness no presenta grandes variaciones luego de la generación 400 ademas el tiempo de ejecución real esta dentro del margen pautado.

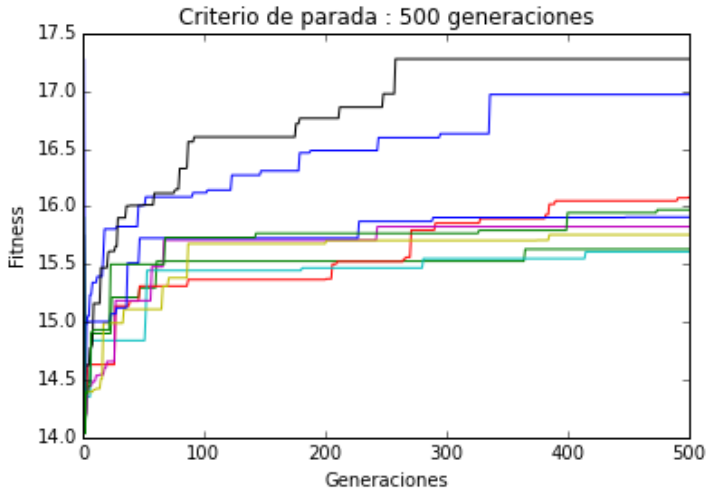

Figura 6.1: Resumen representativo de ejecuciones del algoritmo para establecer el criterio de parada.

6.2.3. Tamaño de la población

Para la elección de la población se tendrán en cuenta 3 elementos. El valor de fitness encontrado, el tiempo de ejecución total y la plataforma de ejecución.

Dado que estamos ejecutando en el cluster y la máxima cantidad de procesadores que se pueden utilizar son 64 en un mismo nodo y teniendo en cuenta que la mejor distribución del trabajo es un elemento de población por procesador, se tiene que la máxima cantidad de población que estudiaremos sera 64.

Luego se eligen los valores 32 y 48 para completar el análisis teniendo en cuenta que no son lo suficientemente bajos como 8 u 16 y son valores con los que se obtiene una distribución más adecuada.

La siguiente tabla muestra los resultados obtenidos, como se aprecia no existen grandes diferencias en la elección de un número poblacional sobre otro. Por tanto se elige como número de población 32 teniendo en cuenta el tiempo de ejecución secuencial del algoritmo que como se aprecia es el menor. Se elige esta métrica por que aunque al ejecutar en paralelo el tiempo real que se puede obtener utilizando la máxima cantidad de

procesadores para cada población son similares también se tiene en cuenta la disponibilidad y utilización de recursos que insume en el cluster fing. Por ejemplo obtener 64 procesadores para utilizar por un proceso en el cluster es algo que puede demorar varios días por la cantidad de otros procesos que también están funcionando en la plataforma y como se ve la ganancia que tenemos es mínima.

Cuadro 6.1: Comparación de fitness para distintas poblaciones

población	**Fitness**		
	mejor	promedio	tiempo de ejecución(m)
32	**17.28**	16.37±0.5	4853
48	**16.19**	15.84±0.3	6772
64	**17.27**	16.46±0.6	10184

6.2.4. Probabilidad de mutación y cruzamiento

Los valores elegidos fueron:

- Probabilidad de cruzamiento (pc): 0.5, 0.8, 1
- Probabilidad de mutación (pm): 0.01, 0.05, 0.1

Se realizan tres juegos de trafico diferente (bajo, medio y alto) y se realizan tres pruebas sobre cada uno, luego se ejecutan las nueve combinaciones de cruzamiento y mutacion obteniendo el promedio del fitness para cada combinacion con su desviacion estandar para realizar la tabla.

Cuadro 6.2: Combinaciones de probabilidad de cruzamiento(pc) y de mutacion (pm)

Combinación(pc-pm)	Fitness Promedio
0.5 - 0.01	16.09±0.30
0.5 - 0.05	15.60±0.17
0.5 - 0.1	16.16±0.42
0.8 - 0.01	16.04±0.55
0.8 - 0.05	15.85±0.32
0.8 - 0.1	16.08±0.34
1 - 0.01	16.08±0.45
1 - 0.05	15.82±0.34
1 - 0.1	16.04±0.25

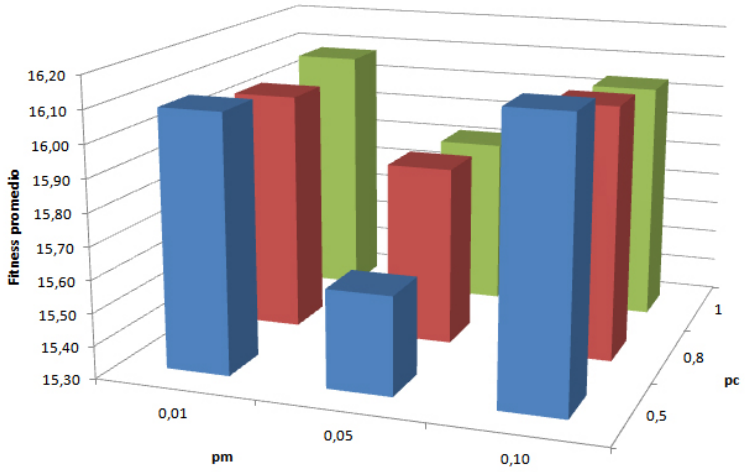

Figura 6.2: Gráfica con combinaciones de probabilidad de cruzamiento(pc) y de mutación (pm)

Analizando la tabla y la grafica se puede apreciar claramente que para una probabilidad de mutacion de 0.05 se obtienen los peores resultados. Otro dato interesante es que no existe gran diferencia en el resto de las combinaciones por lo que vamos a elegir

dos que anuestro criterio son la mejor eleccion y compararlas usando un test parametrico para detectar si existen diferencias significativas entre una y otra.

Se comprueba que todas las muestras siguen la distribución normal para poder aplicar el test de Student.

Siendo las hipótesis de la prueba (u1 el promedio del grupo 1 y u2 el del grupo 2) H0 : u1 = u2 H1: u1 ¡¿u

Si hacemos una comparación entre la combinación de mejor promedio (0.5-0.1) y de peor (0.5-0.05) con el test de Student obtenemos t(x) = 0.07 que nos indica que para un nivel de significancia de 0,1 la hipótesis nula es rechazada por tanto existe evidencia estadistica significativa de que las muestras son diferentes. Por tanto existe una mejora entre elegir una combinación u otra.

Para elegir la mejor opción se toman las dos combinaciones con el mejor promedio (0.5-0.1) y (0.5-0.01) obteniendo en el test Student t(x) = 0.71 que nos indica que no existe una diferencia significativa entre ambas muestras por lo que elegir una sobre otra no implicaría grandes beneficios.

En tal sentido podríamos elegir cualquiera de las dos, en este caso se elige la combinación (0.5-0.01) por su buen promedio y baja desviación estándar.

6.3. Descripción de escenarios

En todos los escenarios se utiliza los datos obtenidos del tráfico vehicular, configuración de semáforos y frecuencia de los ómnibus obtenidos.

6.3.1. Caso base

Esto representa la situación actual en términos de tráfico, red vial y sincronización de semáforos del corredor Garzón.

Se valida su correctitud comparando los tiempos obtenidos en la simulación con tiempos obtenidos in-situ de los recorridos de ida y vuelta para los vehículos. Y utilizando las frecuencias de acceso publico en el caso de los ómnibus.

6.3.2. Escenario Evolutivo

En este caso se ejecuta el algoritmo evolutivo sobre el caso base para obtener una nueva sincronización de semáforos optimizada que repercutirá en la calidad del tráfico.

6.3.3. Escenario Alternativo

Luego de analizar aquellos puntos que se entienden podrían atentar contra el buen funcionamiento del Corredor, se agregan algunas modificaciones al escenario base para intentar mejorarlo.

-Limitar cruces a la izquierda

-Eliminar paradas

-Agregar calles paralelas a Garzon

-

6.4. Resultados

Presentaremos los resultados obtenidos utilizando los parámetros óptimos para el escenario inicial, el escenario modificado, y la prueba en el cluster.

6.4.1. Resultado simulación caso base

El caso base se construye a partir del modelo de tráfico con los datos tomados in-situ por lo que muestra valores aproximados de la realidad actual.

En este caso tenemos las siguentes metricas

Tiempo de viaje promedio de ida en un vehiculo Tiempo de viaje promedio de ida en omnibus por el corredor: Velocidad promedio de los vehículos: Velocidad promedio de los omnibus:

Estas metricas seran las que utilizaremos para comparar con los resultados luego de aplicar el algoritmo.

6.4.2. Resultado Escenario Evolutivo

6.4.3. Resultado Escenario Alternativo

6.4.4. Comparación caso base vs Algoritmo Secuencial

Análisis comparativo : test paramétrico H1) los resultados de mejor fitness tienen una distribución normal. H2)Existe una diferencia significativa entre los de conjuntos de muestras obtenidos por el algoritmo y la realidad.

El test de normalidad de Shapiro-Wilks resultó ser verdadero en ambos casos con un alto porcentaje de confiabilidad

6.4.5. Comparación Secuencial vs Paralelo

El speedup se define como T1/Tp T1 es el tiempo de ejecucion en un procesador Tp es el tiempo de ejecucion en p procesdores.

Vamos a comprobar la utilidad de tener un algoritmo paralelo comparando el tiempo de ejecucion en diferentes número de procesadores

La escabilidad de un algoritmo define su capacidad para aumentar su rendimiento a medida que se agregan más procesadores.

En este caso comparamos el algoritmo ejecutado en 1, 8, 16, 32 procesadores.

GRAFICA DE ESCABILIDAD

El comando "time.^{ej}ecutado sobre nuestro algoritmo nos brinda el tiempo real de ejecución, así como el tiempo total sumado de todos los procesadores. Por tanto podemos calcular el tiempo del algoritmo secuencial.

En el caso de 8 procesadores, al tener una población de 32 se distribuirán 4 individuos por procesador.

Por tanto cuando tenemos 32 es donde se logra la mejor distribución al no existir .^{o}verhead.^{en} la creación y terminación de los hilos.

6.5. Resumen

Capítulo 7

Conclusiones y trabajo futuro

7.1. Conclusiones

A pesar de que el problema de sincronización de semáforos es un problema difícil de abordar, los resultados obtenidos muestran la capacidad de los algoritmos genéticos para resolver problemas de este tipo, obteniendo resultados muy buenos.

El enfoque multiobjetivo aún siendo básico dio buenos resultados en el sentido de priorizar un tipo de trafico u otro.

El desarrollo de algoritmos con capacidad de paralelización son fundamentales sobre todo en problemas complejos que requieren mucho poder de computo como el que se abordo. Y demuestran que son útiles en acelerar el procesamiento.

7.2. Trabajo futuro

La elaboración de los mapas para la simulación donde se incluye cambios en las rutas para que sean reconocidas por el simulador, agregado de la configuración de semáforos, agregado de las lineas y paradas de ómnibus puede ser un proceso lento y tedioso, por tanto para un futuro se podría sugerir la realización de herramientas que automaticen o agilicen este trabajo.

Los trabajos de (Montana and Czerwinski, 1996) y (Vogel et al., 2000) proponen la adaptabilidad del algoritmo en tiempo real, aunque esto requiere del agregado de sensores a la red puede ser un método de mejora importante sobre todo en zonas de gran densidad de trafico.

Bibliografía

Corsim. http://www-mctrans.ce.ufl.edu/featured/TSIS/Version5/corsim.htm.

Transyt-7f. http://mctrans.ce.ufl.edu/mct/?page_id=943.

Autoanuario. Gráficas y cuadros de valores estadísticos del mercado nacional. http://www.autoanuario.com.uy/index_mercado_graficas.html, 2014.

BBVAResearch. Uruguay situación automotriz. https://www.bbvaresearch.com/wp-content/uploads/2014/05/1303_SitAutomotrizUruguay_2013.pdf, 2013.

CEPAL. Congestión de tránsito, el problema y cómo enfrentarlo. http://www.cepal.org/publicaciones/xml/9/13059/cue-87.pdf, 2003.

CHC. Chc method. http://neo.lcc.uma.es/mallba/easy-mallba/html/algorithms.html#chc.

DiFebbraro, Giglio, and Sacco. On applying petri nets to determine optimal offsets for coordinated traffic light timings. *Proc. IEEE 5th Int. Conf. Intell. Transportation Syst.*, pages 773–778, 2002.

J. J. K. G. Y. Lim and Y. S. Hong. The optimization of traffic signal light using artificial intelligence. *10th IEEE Int. Conf.Fuzzy Syst*, page 1279–1282, 2001.

IMM. Intendencia municipal de montevideo. http://www.montevideo.gub.uy.

INE. Transporte y comunicaciones. http://www.ine.gub.uy/biblioteca/uruguayencifras2014/Uruguay_en_cifras_2014_Cap_11.pdf, 2014.

S. López, P. Hernandez, A. Hernandez, and M. Garcia. Artificial neural networks as useful tools for the optimization of the relative offset between two consecutive sets of traffic lights. *Foundations and Tools for Neural Modeling*, page 795–804, 1999.

Mallba. Mallba library. http://neo.lcc.uma.es/mallba/easy-mallba/index.html.

Malva. The malva project. http://themalvaproject.github.io/.

MalvaAlgoritmo. Algoritmo genetico de malva. https://github.com/themalvaproject/malva/wiki/Algorithms:-Genetic-Algorithm.

D. J. Montana. Strongly typed genetic programming. *Evolutionary Computation*, 3(2): 199–230, 1995.

D. J. Montana and S. Czerwinski. Evolving control laws for a network of traffic signals. 1996.

S. Nesmachnow. Evolución en el diseño y clasificación de algoritmos genéticos paralelos,. 2002.

Observador. Imm sincronizará 400 semáforos para agilizar el tránsito capitalino. http://www.elobservador.com.uy/noticia/292066/imm-sincronizara-400-semaforos-para-agilizar-el-transito-capitalino/.

OSM. Open street map. www.openstreetmap.org/, a.

OSM. Osm stats. http://osmstats.neis-one.org/, b.

L. Papaleontiou. Sumo traffic modeler. http://sourceforge.net/projects/trafficmodeler/.

J. Penner, R. Hoar, and C. Jacob. Swarm-based traffic simulation with evolutionary traffic light adaptation. 2002.

A. K. Rathi. Urban network traffic simulation:traf-netsim program. *Journal of Transportation Engineering*, 116(6):734–743, November-December 1990.

N. M. Rouphail, B. B. Park, and J. Sacks. Direct signal timing optimization: Strategy development and results. 2000.

J. Sánchez, M. Galán, and E. Rubio. Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization. 2004.

J. Sánchez, M. Galán, and E. Rubio. Applying a traffic lights evolutionary optimization technique to a real case: "las ramblas" area in santa cruz de tenerife. 2008.

J. Sánchez, M. Galán, and E. Rubio. Traffic signal optimization in "la almozara" district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing. 2010.

D. H. Stolfi. Optimizacion del trafico rodado en ciudades inteligentes. 2012.

Subrayado. Expertos en tránsito: "el colapso está establecido" en montevideo. http://www.subrayado.com.uy/Site/noticia/23835/expertos-en-transito-el-colapso-esta-establecido-en-montevideo, 2013.

SUMO. Simulation of urban mobility. http://sumo.dlr.de/wiki/Main_Page, a.

SUMO. Sumo output. http://sumo.dlr.de/wiki/Simulation/Output, b.

K. T. K. Teo, W. Y. Kow, and Y. K. Chin. Optimization of traffic flow within an urban traffic light intersection with genetic algorithm. 2010.

A. Vogel, C. Goerick, and W. von Seelen. Evolutionary algorithms for optimizing traffic signal operation. 2000.