

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 Computacion - Facultad de Ingenieria

Universidad de la Republica

Montevideo, Uruguay, diciembre de 2014

# ALGORITMOS EVOLUTIVOS EN SINCRONIZACIÓN DE SEMAFOROS EN EL CORREDOR DE GARZÓN

## RESUMEN

Este trabajo estudia la aplicación de un algoritmo evolutivo multiobjetivo la resolución de problemas de sincronización de semáforos en un escenario real. Este tipo de problemas son NP-HARD es decir que no existe un algoritmo que lo resuelva en tiempo polinómico. Los algoritmos evolutivos han sido aplicados con éxito a este tipo de problemas. Luego de una inversión que se asume fue de aproximadamente 33 millones de dólares y una infraestructura que pretendía ser de primer mundo, no se pudo agilizar el transporte público lo cual era el único objetivo que tuvo desde un principio en la obra llamada “Corredor de Garzón”. Se pretende en este trabajo mejorar mediante algoritmos evolutivos la sincronización de los semáforos, para poder lograr agilizar el transporte, minimizando los tiempos de espera. Presentaremos los resultados y tiempos de ejecución del algoritmo y compararemos con los datos recabados inicialmente, con los obtenidos en el algoritmo utilizando el simulador de tráfico SUMO en distintos escenarios

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



# Índice general

<b>1. Introducción</b>	<b>1</b>
<b>2. Marco Teórico</b>	<b>3</b>
2.1. Planificación del trafico . . . . .	3
2.2. Algoritmos Evolutivos . . . . .	3
2.2.1. Introduccion . . . . .	3
2.2.2. Algoritmos Evolutivos . . . . .	3
2.2.3. Algoritmos Geneticos . . . . .	3
2.2.4. Algoritmos Genetico Simple (AGS) . . . . .	3
2.2.5. Algoritmos Genetico MultiObjectivo . . . . .	3
2.2.6. Algoritmo Genetico Paralelo . . . . .	3
2.2.7. Maestro-Esclavo . . . . .	3
2.2.8. Resumen . . . . .	3
2.3. Por que usar algoritmo geneticos? . . . . .	3
2.4. Simulación y herramientas . . . . .	3
2.4.1. Simuladores de trafico . . . . .	3
2.4.2. Modelo de trafico . . . . .	4
2.4.3. Representaciones . . . . .	4
2.4.4. Ajustes de archivos de SUMO . . . . .	5
2.4.5. Trafico . . . . .	5
2.4.6. Trabajo de campo realizado . . . . .	5
2.4.7. Configuración de la simulación . . . . .	5
2.5. Resumen . . . . .	6
<b>3. Presentación y trabajos relacionados</b>	<b>7</b>
3.1. Presentación . . . . .	7
3.2. Formulación matemática . . . . .	7
3.3. Estado del arte . . . . .	7
3.4. Resumen . . . . .	11
<b>4. Estrategia de resolución</b>	<b>13</b>
4.1. Algoritmo Genetico . . . . .	13
4.1.1. representacion . . . . .	13
4.1.2. Codificacion . . . . .	13
4.1.3. inicializacion . . . . .	14
4.1.4. funcion fitness . . . . .	14
4.1.5. operadores . . . . .	14

4.1.6. Criterio de parada . . . . .	14
4.2. Algoritmo Paralelo . . . . .	14
4.3. Resumen . . . . .	15
<b>5. Análisis Experimental</b>	<b>17</b>
5.1. Descripción de escenarios . . . . .	17
5.2. Desarrollo y plataforma de ejecucion . . . . .	17
5.3. Ajuste de parametros de algoritmos . . . . .	17
5.4. Resultados . . . . .	17
5.4.1. Resultado simulacion caso base . . . . .	18
5.4.2. Resultado Escenario Inicial ( Algoritmo Secuencial) . . . . .	18
5.4.3. Resultado Escenario Modificado ( Algoritmo Secuencial) . . . . .	18
5.4.4. Resultado de ejecucion en paralelo. . . . .	18
5.4.5. Comparacion caso base vs Algoritmo Secuencial . . . . .	18
5.4.6. Comparacion Algoritmo Secuencial vs Algoritmo Paralelo (Calculo de SpeedUp) . . . . .	18
5.4.7. Escabilidad de Algoritmo Paralelo . . . . .	18
5.5. Resumen . . . . .	18
<b>6. Conclusiones y trabajo futuro</b>	<b>19</b>
6.1. Conclusiones . . . . .	19
6.2. Trabajo futuro . . . . .	19
<b>Bibliography</b>	<b>19</b>

# Índice de figuras





# Índice de cuadros



# List of Algorithms



# Capítulo 1

## Introducción

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

### **Motivación y contexto**

Con la idea de agilizar el transporte público entre los barrios Colón y Sayago por un lado y la Ciudad Vieja, Pocitos y Malvín por otro, barrios que son bastante lejanos el uno del otro en la ciudad de Montevideo Uruguay, se realiza una obra cuya intención radica en que la clase obrera demorará menos tiempo en llegar a su destino. Supuestamente se iban a acortar los tiempos del transporte público de la hora que llevaba ir de Colón a Ciudad Vieja por ejemplo a aproximadamente 25 minutos lo cual no se pudo cumplir y aunque es exagerada esa posible reducción de tiempo, si podría cumplirse algo más plausible. En este momento aun estando en una calle cuya cartelería permite circular a 60 Km/h [1], se tiene una velocidad promedio de 25 Km/h (según varios viajes realizados) lo cual tiene muchísimo para mejorar. La intendencia luego de diversas protestas de la gente que vivía en los barrios por los que pasa el corredor de Garzón da marcha atrás a proyectos similares como el corredor Agraciada Norte[2], y General Flores, admitiendo que en Garzón no se había cumplido el objetivo[3].

La obra consistió en ensanchar las calles de Garzón, manteniendo las dos vías para cada sentido y agregando un corredor central exclusivo para los ómnibus Urbanos (que no salen de Montevideo), este es el llamado Corredor de Garzón. Garzón maneja un conjunto de 6 de semáforos por cruce, teniendo un cruce cada 300 metros aproximadamente. Se pretende mejorar el estado de la sincronización de los semáforos en una región de Garzón. Junto a esto se intentará minimizar los tiempos de espera que se puedan obtener en el presente

### **Objetivos**

Mostrar que los algoritmos evolutivos pueden solucionar problemas complejos en escenarios reales, siendo una herramienta perfectamente utilizable.

### **limitaciones**

La información exacta sobre la configuración de los semáforos y la densidad de tráfico no está disponible públicamente, por tanto el relevamiento de datos hecho in situ tiene fue realizado para un número determinado de calles y días.

**Enfoque**

Se creara un programa que implemente un algoritmo genetico y llama a un simulador de trafico para obtener las metricas a optimizar.

**Contribuciones de este trabajo****Contacto con el publico**

Cabe destacar que este proyecto se presento en Ingeniera de muestra, siendo bien recibido por el publico. Donde constatamos de primera mano que la problematica es real y nosotros como Ingenieros tenemos las herramientas necesarias para solucionar problemas que afectan directamente a la sociedad.

**Estructura del documento**

El capitulo 2 brinda el marco teorico necesario para poder comprender los siguientes capitulos sobre la implemmentacion del algoritmo y las simulaciones. Ademas que se da un pantallazo al problema del trafico. El capitulo 3 mostramos los trabajo relacionados haciendo incapie en algoritmos geneticos para la sincronizacion de semaforos. el capitulo 4 explicamos la estrategia seguida para la resolucion del problema y damos en detalle el disenio de la misma. El capitulo 5 cuenta con tablas, graficas e informacion relacionada con el analisis experimental realizado en los distintos escenarios que elegimos. El capitulo 6 da las conclusiones finales y el trabajo a futuro que se puede realizar.

## Capítulo 2

# Marco Teórico

En este capitulo se aborda el marco teorico necesario para comprender mas facilmente el desarrollo de los capitulos posteriores. Se analiza el problema del trafico en general, los simuladores y los algoritmos a utilizar.

### 2.1. Planificación del trafico

### 2.2. Algoritmos Evolutivos

#### 2.2.1. Introduccion

#### 2.2.2. Algoritmos Evolutivos

#### 2.2.3. Algoritmos Geneticos

#### 2.2.4. Algoritmos Genetico Simple (AGS)

#### 2.2.5. Algoritmos Genetico MultiObjectivo

#### 2.2.6. Algoritmo Genetico Paralelo

#### 2.2.7. Maestro-Esclavo

#### 2.2.8. Resumen

### 2.3. Por que usar algoritmo geneticos?

El problema de sincronización de semáforos es NP-Hard y que 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 de la región más transitada del corredor de Garzón.

### 2.4. Simulación y herramientas

#### 2.4.1. Simuladores de trafico

Los simuladores de trafico son programas que simulan el movimiento de vehiculos sobre una red de calles,es una herramienta muy usada en la investigacion de trafico

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 rápido que en la realidad, el costo en dinero ya que 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 es uno de los simuladores abiertos más populares, es microscópico aunque presenta algunas dificultades a la hora de la configuración del mapa y el tránsito que lo hace en base a archivos XML.

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 acondicionamiento de estos archivos. SUMO simula el tráfico utilizando archivos XML que representan las rutas, los vehículos y el tráfico. En

#### **2.4.2. Modelo de tráfico**

Esta es la representación de la circulación de vehículos, existen varios métodos desde Aleatorios: Genera diferentes recorridos que seguirán los vehículos aleatoriamente JTR : basados en probabilidades en los cruces es decir cuando un vehículo llega a un cruce tiene cierta probabilidad de seguir o doblar (JTR – junction turning ratio), Basado en distritos: Se especifican distritos(conjunto de calles) y la cantidad de movimiento vehicular entre ellos en una matriz 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.

#### **2.4.3. Representaciones**

##### **Red calles**

La red de calles se representa como un gráfico 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 trips. Un trip representa el movimiento de un vehículo de un punto inicial hacia un punto final (El recorrido se hace en tiempo de ejecución utilizando el camino más corto basado en el tráfico). Una ruta es más compleja que un trip ya que agrega todos los vértices por los que el vehículo pasará.



SUMO tambien soporta el modelo JTR y basado en discretos pero se necesitan modulos externos para generarlos.

### **Representacion 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 asi para que sea concistente

#### **2.4.4. Ajustes de archivos de SUMO**

Pasos y problemas en la generacion y adecuacion del mapa

#### **2.4.5. Trafico**

Elegimos utilizar el flow calle-a-calle que nos permite determinar con mucho detalle el trafico generado entre calles. Ya que contamos con datos relevados en el lugar. Se especifica el lugar de donde sale el vehiculo, donde termina su recorrido, el numero de vehiculos que se emiten o la frecuencia.

### **Tipos de vehiculos**

Se pueden crear diferentes tipos de vehiculos especificando propiedades como largo, velcoidad maxima, aceleracion, color, etc. Tambien contamos con algunos por defecto como camiones, buses

### **Accidentes**

El simulador permite representar colisiones y el corte de una calle. Decidimos no utilizar esto pues no queriamos este tipo de variables afectara en la ejecucion de las pruebas.

#### **2.4.6. Trabajo de campo realizado**

Al no contact con datos publicos sobre la configuracion de los semaforos de la zona, realizamos un relevamiento in situ de la siguiente forma. En los principales cruces realizamos una filmacion de 30 min. Luego analizamos el video realizando el conteo manualmente con la posibilidad de enlenter el video para mayor facilidad. Estas medidas nos sirvieron para verificar nuestros modelos basados en los datos del GPS

La configuracion de los semaforos se realizo yendo por el corredor y cronometrando la duracion del tiempo. Tanto en ida como en venida para coprobarar que fueran correctos.

#### **2.4.7. Configuración de la simulación**

##### **Diseño del mapa**

El mapa base de la zona lo tomamos de OSM, luego se cotejo su exactitud con Google Maps y Bing Maps. Utilizamos la herramienta netconvert para pasarlo a formato que SUMO acepte. Para esto realizamos varios ajustes editando los archivos xml para indicar donde se realizan.

Se generó una especial dificultad en el diseño de un mapa que fuese tan fiel como fuese posible a la realidad y que también fuese compatible para usar con Sumo. Se debió recabar datos in situ ya que hasta la misma intendencia [4] tenía errores respecto a la realidad, falta de algunos semáforos, etc. Luego al pasarlo al formato compatible con sumo se debieron corregir calles, cruces y todas las posibles conexiones de las esquinas del corredor que debieron ser escritas a mano en un XML.

Poner fotos de un ants (salida directa del netconverte) y luego con los agregados de joins y connections

### **Vehículos**

Se manejaron dos tipos de vehículos, autos y ómnibus: Las líneas de ómnibus urbanas (que circulan por el corredor) incluidas fueron la “G”, la “409” y adaptaciones de la “522” y “148” que se juntaron en una línea sola. Y las líneas de ómnibus suburbanas en el tramo elegido realizan el mismo trayecto y les llamamos línea “A”. Todas las líneas de ómnibus fueron cargadas con las paradas correspondientes y se hicieron variantes en los viajes dentro de una misma línea de manera que no siempre paran en los mismos lugares. Los viajes de los autos son generados aleatoriamente de modo que no circulen por el corredor (que es para los ómnibus urbanos) y de forma que tengan mayor probabilidad los viajes que comienzan y terminan en el borde de la red, luego se seleccionan los viajes de manera que las grandes avenidas sean más recorridas que las calles menos importantes.

### **Semáforos en cada cruce**

Se tomaron los tiempos de cada luz de los semáforos en cada cruce y luego estos datos fueron cargados al simulador para comparar con los resultados así como también para base de nuestro algoritmo evolutivo.

### **Escenarios**

## **2.5. Resumen**

## Capítulo 3

# Presentación y trabajos relacionados

### 3.1. Presentación

### 3.2. Formulación matemática

### 3.3. Estado del arte

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

El problema del tráfico optimizando las luces de los semáforos se puede resolver por muchos metodos como autómata celular, redes neuronales, fuzzy logic, redes de petri, sistema expertos o programación lineal por lo tanto la cantidad de soluciones encontradas fue abundante y variada por esto decidimos enfocarnos en soluciones lo más cercanas a la nuestra posible y en otras que tuvieran alguna particularidad interesante que nos gustaría destacar.

- SÁNCHEZ, GALÁN, and 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 esta 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, además usa mutación variable dependiendo.

Para la evaluación se usa el tiempo medio, esto es desde el momento que un vehículo entra en la red hasta que sale. Se utilizó un cluster y programación paralela utilizando MPI 2 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 simulación fija, obteniendo la solución propuesta mejores resultados en todos los casos evaluados. Este mismo grupo realizó trabajos similares expandiendo esta investigación, estos son:

- SÁNCHEZ, GALÁN, and 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 como código Gray lo que 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 utilizó para comparar con los resultados obtenidos por el algoritmo que en términos generales logra un aumento del rendimiento de hasta 26.

- SÁNCHEZ, GALÁN, and 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.
- SÁNCHEZ. Estudio de la optimización del tráfico rodado mediante el ajuste de los ciclos de semáforos por algoritmos genéticos en dispositivos de computación paralela usando modelización discreta: Ejemplos de aplicación. 2007 En esta tesis se conjugan varios de sus trabajos que ya comentamos, ampliando y profundizando en varios puntos.

- 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 swarms (enjambres) utilizando el programa SurJe para el mapa y la simulación. 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 relación con los semáforos complementarios, la mutación tiene en cuenta esto para que no ocurra en una misma intersección 2 luces verdes. El cruzamiento se hace entre los distintos semáforos con una probabilidad más alta si está 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 mas complejo de 28 semaforos 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 inteligente ya que indica que los atascos del trafico provocan no solo perdidas economicas sino tambien contaminacion ambiental.

Para ello propone utilizar un algoritmo inteligente que tomando en cuenta el estado de congestion de las rutas sugiere al usuario cual es la ruta mas rapida a su destino, utilizando un dispositivo en el automovil que se enlazara por wifi con los semaforos (que cuentan con sensores). Por lo tanto el trabajo no se basa en la optimizacion de las señales de los semaforos existentes sino agrega encima de esto un sistema de busqueda de mejor ruta.

Para el modelo utiliza una zona de la ciudad de malaga obtenido desde Open Street Map, cuenta con 8 entradas y 8 salidas , para la simulacion utiliza SUMO. Los vehiculos modelados son : turismo, monovolumen, furgoneta, camion donde se varia la longitud, velocidad y probabilidad que entre en la red de trafico.

Se intenta minimizar los tiempo de viaje de los vehiculos que circulan por la red. Para ellos se utiliza un algoritmo genetico cuya estrategia de seleccion concisten en tomar los 2 peores individuos y reemplazandolo por los 2 mejores hijos encontrados. En el cromosoma se representa cada sensor, con los destinos y rutas posibles. La funcion de fitness tiene en cuenta la cantidad de viajes completados durante el tiempo de ejecucion, el tiempo medio utilizado, y el retraso medio. Se prueban varias estrategias de cruzamiento y mutacion. Las ejecuciones tienen un tiempo fijo de duracion.

Compara el resultado con una simulacion por defecto realizada con el programa DuaRouter que viene con SUMO donde se generaron 64 itinerarios diferentes, esto se prueba en 3 escenarios diferentes. Las simulaciones se realiza hata con 800 vehiculos,se concluye que al aumentar la cantidad de vehiculos (mas de 400) en el sistema la solucion 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 interseccion en donde se intenta optimizar los tiempos de los semaforos para lograr mejor rendimiento.El cromosoma representa los tiempos de la luces verdes, el cruzamiento toma 80 de informacion de un padre y 20 del otro. La funcion de fitness es el largo de las colas generadas. La simulacion tiene un tiempo fijo de 600 segundos por generacion pero no se detalla el tipo que se utilizo. Las conclusiones indican que la optimizacion usando algoritmos geneticos es buena para el problema del flujo de trafico.
- D. J. Montana and S. Czerwinski. Evolving control laws for a network of traffic signals. 1996 Utiliza un enfoque adaptativo con sensores que analizan el trafico en tiempo real (un sensor para saber cuantos autos pasan y otro para saber que tan larga es la cola) tomando en consideracion 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 será ejecutado por todas las intersecciones cuando decida el cambio de fase. Además un algoritmo genético híbrido busca diferentes constantes que serán usadas en los árboles de decisión, permitiendo una especialización en las diferentes geometría y 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. El simulador usado utiliza una versión especial de TRAF-NETSIM.

El experimento arroja buenos resultados en cuanto a la performance de la red comparando con un ciclo fijo, y que presenta buena adaptabilidad en diferentes circunstancias. Marca el hecho de que el modelo es simple y de tamaño pequeño, y que es 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 autoadaptable 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 señalización aun siendo optimizada usando simulaciones es difícil que sea la mejor en todas las situaciones o en casos extremos (horas pico). Para solucionar esto proponen un sistema autoadaptable que toma la información del tráfico actual usando detectores de vehículos y de 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. Utiliza algoritmos evolutivos donde cada individuo representa un sistema de fases junto con sus parámetros. Usa. El fitness se obtiene simulando ese sistema en un modelo de tráfico. Este modelo es relativamente pequeño una intersección con 4 brazos, cada uno con 3 líneas una de ellas para doblar a la izquierda, la del medio para ir derecho, y la restante para giros a la derecha. La ruta principal tiene el doble de densidad vehicular que la que la cruza.

El simulador utilizado está basado SIMVAS++, (Technische Universität Dresden, Fakultät für Verkehrswissenschaften (Ringel (1995)))

Los resultados indican que la ventaja de usar conocimiento experto para inicializar parámetros 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 con 9 intersecciones con semáforos en la ciudad de Chicago (USA), contando con parking, rutas de omnibus y paradas así

como con vehiculos. Se toman valores reales en horas pico AM y PM, comprobando que las colas que se generan en la simulacion coinciden con la realidad. Usa el programa TRANSYT-7F ( Que permite visualizar mapas y contiene optimizacion de varios algoritmos geneticos. Se probaron 12 estrategias distintas en 7F y la mejor fue simulada en CORSIM 100 veces. Se midio el tiempo de demora en la red y el largo de las colas producidas La performance de la red aumentó considerablemente usando este metodo

### 3.4. Resumen

Como se aprecia analizando el estado del arte nuestraa implementación es mucho más compleja que los t





## Capítulo 4

# Estrategia de resolución

### 4.1. Algoritmo Genetico

El algoritmo que vamos a utilizar es el algoritmo simple, este sigue la propuesta de Goldberg [5]. El algoritmo en galib se llama GASimpleGA, el mismo crea una población inicial y en cada generación, genera una nueva población de individuos seleccionando de la población anterior y realizando cruzamiento, hace un reemplazo total. El proceso se repite hasta dar con algún tipo de criterio de parada. La librería se basa en el elitismo, los mejores individuos son llevados hacia la siguiente generación.

#### 4.1.1. representacion

Para poder explicar la representación del problema de los semáforos del corredor Garzón, es preciso compartir algunas definiciones que usaremos a lo largo de la propuesta: Cruce: se define como 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; un cruce tiene hasta 8 fases. Por ejemplo una fase de un cruce puede ser “rrrrGGrrrrGG” durante 52 segundos. Donde “G” es Verde, “r” es Rojo y “y” es Amarillo. Es importante que el algoritmo evolutivo a implementar no genere soluciones que no sean viables por lo que éste no debe de poder modificar la combinación de luces de cada fase y de

esta manera no se generarán fases con combinaciones de luces equivocadas (podría generar accidentes de tránsito). En pos de un mejor rendimiento y ya que en realidad no modifican los tiempos reales del paso de los vehículos, se omitieron las fases que tienen luces amarillas en la representación del cromosoma. El cromosoma se va a agrupar lógicamente en cruces, definimos el valor de un gen como el tiempo que demora una fases de un cruce (sin considerar las amarilla, estas quedan fijas con el valor obtenido en la realidad) o la fase con la que inicia un cruce, por lo que el tamaño del cromosoma depende de la cantidad de cruces y de la cantidad de fases que tiene cada cruce, teniendo así la información de todos los cruces.

#### 4.1.2. Codificacion

A continuación se muestra un ejemplo en el cual se mapea un cruce representado en el cromosoma y un cruce representado en el archivo de configuración de semáforos que utiliza el simulador SUMO. Como se mencionó anteriormente se omite el valor de la luz amarilla en el cromosoma y se mantiene el valor real recabado y el valor que tiene el

cromosoma en el gen de inicio de fase se corresponde con el valor de “offset” en el XML. Cruce representado en el cromosoma:

#### 4.1.3. inicializacion

Para la inicialización de la población se toma como referente 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 segundos a 60 segundos (estos valores son configurables) y la fase inicial se hace variar aleatoriamente entre la cantidad de fases del cruce (se cuentan las luces amarillas).

#### 4.1.4. funcion fitness

Para evaluar un individuo, lo que se hace es crear un archivo XML de configuración de señalización de tránsito que utiliza el simulador SUMO en base al cromosoma del individuo, luego se ejecuta el simulador con la configuración generada, el tiempo de simulación devuelto será el valor de la función de fitness.

#### 4.1.5. operadores

##### seleccion

##### cruzamiento

Se utilizará cruzamiento de un punto (1PX), implementado específicamente para el problema, seleccionando el intervalo entre 2 cruces como punto de corte, por ejemplo: Esto hace que si un tramo del corredor es bueno, esta propiedad se mantenga.

##### mutacion

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.

#### 4.1.6. Criterio de parada

Cantidad de generaciones: Es la cantidad de generaciones con la cual se toma el criterio de parar la ejecución del algoritmo, el número de generaciones que vamos a usar es 100. Probabilidad de convergencia: La convergencia se define como la relación entre los N mejores valores de fitness de las generaciones anteriores contra el mejor valor de la generación actual. Nosotros vamos a utilizar N con el valor de 10 y una probabilidad de convergencia de 0.99

## 4.2. Algoritmo Paralelo

Agregar diagrama de maestro esclavo

### 4.3. Resumen



## Capítulo 5

# Análisis Experimental

En esta seccion describimos los distintos escenarios que vamos a probar y los resultados obtenidos.

### 5.1. Descripción de escenarios

### 5.2. Desarrollo y plataforma de ejecucion

Los algoritmos fueron desarrollados usando la librería Malva que fue extendida en el codigo base para soportar la creacion de nuevos hilos de ejecucion para lograr el funcionamiento en paralelo

### 5.3. Ajuste de parametros de algoritmos

Buscamos la mejor configuracion inicial de los parametros realizando pruebas experimentales con diferentes combinaciones. Estos son: el tamaño de la poblacion, probabilidad de mutacion, probabilidad de cruzamiento, etc. Para esto se realizaron 20 ejecuciones independientes para el algoritmo secuencial inicial. El criterio de parada se eligio por el numero de generacion. . .

Para la poblacion se probaron 20, 50, 100 . Los resultado indicaron que Para el cruzamiento 0.5, 0.8, 1 Para mutacion 0.01, 0.05 y 0.1

La mejor configuracion obtenida fue: Poblacion:120, mutacion:0.01 , cruzamiento: 1

Las graficas muestras el promedio de las 20 ejecuciones para resolver el escenario inicial.

### 5.4. Resultados

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

**5.4.1. Resultado simulacion caso base****5.4.2. Resultado Escenario Inicial ( Algoritmo Secuencial)****5.4.3. Resultado Escenario Modificado ( Algoritmo Secuencial)****5.4.4. Resultado de ejecucion en paralelo.****5.4.5. Comparacion caso base vs Algoritmo Secuencial**

Analisis comparativo : test parametrico 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 y luego al realizar los test T-student [6] resulto tener menos de 0,0001 lo que se considera una diferencia que estadísticamente es extremadamente significativa. Esto confirma algo que resulta un tanto evidente ya que al comparar el promedio de los mejores fitness con la realidad encontramos una diferencia de un 31,8casos son bastante acotados ya que hay mucha cantidad de vehículos con grandes/medias distancias. Con los resultados obtenidos en las ejecuciones para los escenarios 1 y 2 se obtuvo un mejor fitness de 661 y 515 respectivamente, comparado con el tiempo de la configuración real que demora 1019 y 690, las soluciones son muy buenas.

**5.4.6. Comparacion Algoritmo Secuencial vs Algoritmo Paralelo (Calculo de SpeedUp)****5.4.7. Escalabilidad de Algoritmo Paralelo**

Ejecutar en 4, 8 ,16, 32

**5.5. Resumen**

## Capítulo 6

# Conclusiones y trabajo futuro

### 6.1. Conclusiones

A pesar de que el problema de sincronización de semáforos es un problema muy difícil de abordar, los resultados obtenidos muestran la capacidad de los algoritmos genéticos para abordar problemas de este tipo, obteniendo resultados muy buenos. Si bien el algoritmo planteado tiene ciertas limitaciones como por ejemplo es dependiente del tráfico que exista, se pueden hacer estudios sobre el tráfico para tener resultados más adaptados a la realidad. También se podría cambiar el enfoque y plantear el problema como un problema multiobjetivo, cada individuo (que representa la configuración de semáforos) se podría evaluar con distintos flujos de tránsito, pudiendo así definir distintos criterios sobre cómo evaluar los individuos.

### 6.2. Trabajo futuro





# Bibliografía

- D. J. Montana and S. Czerwinski. Evolving control laws for a network of traffic signals. 1996.
- J. Penner, R. Hoar, and C. Jacob. Swarm-based traffic simulation with evolutionary traffic light adaptation. 2002.
- N. M. Rouphail, B. B. Park, and J. Sacks. Direct signal timing optimization: Strategy development and results. 2000.
- SÁNCHEZ. Estudio de la optimizacion del trafico rodado mediante el ajuste de los ciclos de semaforospor algoritmos geneticos en dispositivos de computacion paralela usando modelizacion discreta: Ejemplos de aplicacion. 2007.
- SÁNCHEZ, GALÁN, and RUBIO. Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization. 2004.
- SÁNCHEZ, GALÁN, and RUBIO. Applying a traffic lights evolutionary optimization technique to a real case: “las ramblas” area in santa cruz de tenerife. 2008.
- SÁNCHEZ, GALÁN, and 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.
- 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.