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, trafico y cantidad de semáforos lo hace un problema interesante desde el punto de vista de su complejidad.Ademas es real y admitido por las autoridades responsables de que hubo 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 para un conjunto de escenarios, minimizando los tiempos de espera de los vehículos y mejorando de esta manera la calidad del trafico. 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

List of Algorithms

Capítulo 1

Introducción

> I recall seeing a package to make quotes
>
> Snowball

En esta sección 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. Nuestra intención es desarrollar un algoritmo que logre resolver este problema con buenas métricas. Por la flexibilidad inherente de este algoritmo es que no esta destinado a resolver el problema en una zona en particular sino que se podría aplicar a cualquiera.

En este sentido hemos elegido la zona del corredor Garzón que presenta particularidades que la destacan y la hacen interesante desde el punto de vista de la investigación de nuestro algoritmo. 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 trafico aplicadas a cada tramo como por ejemplo exclusividad del ómnibus, o distinción para doblar a la izquierda, trafico vehicular y transporte publico, calles no paralelas, entre otros.

Por tanto si probamos que el algoritmo funciona en esta zona tan compleja podemos tener confianza de que se puede usar en otras zonas que no presentan tanta complejidad y obtener buenos resultados.

1.0.2. Objetivos

Estos son los objetivos básicos que nos planteamos al inicio del proyecto.

- Estudio del problema del trafico y la sincronización de semáforos.
- Recabar 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, trafico 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. Limitaciones y alcance

La información exacta sobre la configuración de los semáforos y la densidad de trafico no esta disponible públicamente, por tanto el revelamiento de datos hecho in-situ fue realizado para un numero determinado de calles y días.

Lo que se pretende modelar es el trafico vehicular y transporte publico por tanto la simulación de personas y los semáforos peatonales no están incluidos.

1.0.4. Enfoque

Se creara un programa que implemente un algoritmo evolutivo que utiliza un simulador de trafico abierto para obtener las métricas a optimizar.

1.0.5. 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 problemática es real y nosotros como Ingenieros tenemos las herramientas necesarias para solucionar problemas que afectan directamente a la sociedad.

1.0.6. Estructura del documento

El capitulo 2 brinda el marco teórico necesario para poder comprender los siguientes capítulos, se da una introducción sobre los algoritmos evolutivos, el problema del trafico y Garzón particularmente. El capitulo 3 mostramos los trabajo relacionados enfocándonos en algoritmos genéticos para la sincronización de semáforos. el capitulo 4 explicamos la estrategia seguida para la resolución del problema y damos en detalle el diseño de la misma. El capitulo 5 cuenta con tablas, gráficas e información relacionada con el análisis 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 teórico necesario para comprender mas fácilmente el desarrollo de los capítulos posteriores. Se analiza el problema del trafico en general, los simuladores y la teoría detrás del algoritmo a utilizar.

2.0.7. Problema del transito vehicular

En gran parte del mundo se esta 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 .[http://www.cepal.org/publicaciones/xml/9/13059/cue-87.pdf]

Este problema tiene un impacto grande 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. Ademas se genera una exigencia en las vías de transito que ocasiona un deterioro mayor de calles y rutas.

Uruguay no escapa a este fenómeno en particular Montevideo, donde el aumento del parque automotor esta en ascenso constante desde el 2005 Y según proyecciones el crecimiento seguiría en un promedio de 4.5

Esto viene de la mano con el sostenido aumento de las ventas de vehículos desde el 2003

Los expertos indican que la congestión ya esta instalada y la infraestructura vial no acompaso este crecimiento. Ademas se indica que Montevideo es la ciudad con mas semáforos por automóvil en Latinoamerica. Con mas de 620 cruces semaforizados, alguno de los cuales no estan coordinados.[http://www.subrayado.com.uy/Site/noticia/23835/expertos-en-transito-el-colapso-esta-establecido-en-montevideo]

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

2.0.8. Corredor Garzón

El corredor Garzón esta ubicado 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

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

2.1. Algoritmos Evolutivos

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. []

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

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

2.1.1. Algoritmos Genéticos

Un algoritmo genético es un tipo de algoritmo evolutivo, siendo de los mas populares.

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 numero de iteraciones o cuando ya no se puede mejorar mas 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 alcanzo un numero de generaciones predefinidos o por que la mejora no es tan 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. Ruleta: Torneo: Elitismo:

Cruzamiento

Su función es combinar individuos para lograr mejores soluciones. Existe una taza que se puede modificar para indicar la probabilidad de que se realice el cruzamiento. Cruzamiento de un punto: Cruzamiento azar

Mutación

Indica el método utilizado para modificar un individuo, esto se realiza para lograr mas diversidad y no caer en máximos locales. En general aplica una modificación aleatoria en el cromosoma. También hay una taza de probabilidad, en general es baja.

Reemplazo

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

Criterio de parada:

Indica cuando debe terminar el algoritmo, puede ser definiendo un numero fijo de generaciones o analizando si la mejora en soluciones se estanco.

2.1.2. Algoritmos Genético Simple (AGS)

El algoritmo que vamos a utilizar es el simple, propuesto por Goldber [] El operador principal es el de cruzamiento y el secundario la mutación.

Su esquema de funcionamiento es el siguiente

Algorithm 1 Algoritmo Genético Simple

```
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
```

2.1.3. Algoritmos genético multiobjetivo

Explicación

En este caso se busca una solución que satisfaga de forma simultanea las restricciones y varios objetivos diferentes.

Por que elegimos multiobjetivo?

La intendencia apoya el cambio del trafico en la ciudad de Montevideo para que se haga mayor uso del transporte urbano[] y disminuir la cantidad de automóviles circulantes para disminuir el trafico y la contaminación. En este sentido se da prioridad a los ómnibus, por esta razón nuestro algoritmo puede varían los diferentes pesos para dar mas prioridad a la velocidad promedio de ómnibus vs el resto de los vehículos.

Se busca que la gente prefiera el ómnibus, uno de los factores es la duración del viaje, si se logra mejorar esto se estaría cumpliendo uno de los objetivos de la intendencia SUMO brinda mucha información referente a los viajes, por lo que tenemos varias métricas a elegir para nuestro algoritmo. Entre ellas se destacan: Cantidad de vehículos que llegaron a destino, Tiempo de viaje promedio, Tiempo de ocupación promedio, velocidad promedio global, tiempo parado q emite gases. Decidimos usar la velocidad promedio separando entre ómnibus y autos. De esta forma podemos optimizar ambos en el sentido multiobjetivo, o priorizar uno sobre el otro.

2.1.4. Algoritmo Genético Paralelo

2.1.5. Maestro-Esclavo

2.1.6. Resumen

2.2. Simulación y herramientas

2.2.1. Simuladores de trafico

Los simuladores de trafico son programas que simulan el movimiento de vehículos sobre una red de calles,es una herramienta muy usada en la investigación 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 mas rápidos 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. Ademas 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 trafico 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 mas populares, es microscópico aunque presenta algunas dificultades a la hora de la configuración del mapa y el transito que lo hace en base a archivos XML.

Cuanto mas crece el numero de vehículos y la complejidad de la red de mapas mas difícil se hace crear la entrada básica que necesita el simulador. Aunque existen diversas herramientas que ayudan a este proceso aun se requiere un trabajo manual para el acondicionado de estos archivos. SUMO simula el trafico utilizando archivos XML que representan las rutas, los vehículos y el trafico. En

2.2.2. Modelo de trafico

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 trafico 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.

2.2.3. Representaciones

Red calles

La red de calles se representa como un gráfico dirigido en un archivo xml con extensión .net.xml . Alli 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 Trafico

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 hacen en tiempo de ejecución utilizando el camino mas corto basado en el trafico). Una ruta es mas complejo que un trip ya que agrega todos los vértices por los que el vehículo pasara.

SUMO también soporta el modelo JTR y basado en distritos pero se necesitan módulos externos para generarlos.

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.

2.2.4. 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 vehículo, donde termina su recorrido, el numero de vehículos que se emiten o la frecuencia.

Tipos de vehículos

Se pueden crear diferentes tipos de vehículos especificando propiedades como largo, velocidad máxima, aceleración, color, etc. También 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 queríamos este tipo de variables afectara en la ejecución de las pruebas.

2.2.5. Trabajo de campo realizado

Al no contar con datos públicos sobre la configuración de los semáforos de la zona, realizamos un revelamiento in-situ de la siguiente forma. En los principales cruces realizamos una filmación de 30 min. Luego analizamos el vídeo realizando el conteo manualmente con la posibilidad de enlentecer el vídeo para mayor facilidad. Estas medidas nos sirvieron para verificar nuestros modelos basados en los datos del GPS

La configuración de los semáforos se realizo yendo por el corredor y cronometrando la duración del tiempo. Tanto en ida como en venida para coproborar que fueran correctos.

2.2.6. Configuración de la simulación

Generar un mapa compatible con el simulador de transito mejor al que tenemos del año pasado con los últimos detalles del corredor.

Edicion del mapa http://fmachado.dei.uc.pt/wp-content/papercite-data/pdf/misc13.pdf pequeño articulo sobre sumo, mismas dificultades que tuvimos el año pasado esperemos que las mejoras del netconvert hayan sido buenas sino vamos a estar un buen rato tocando xml

Creación del modelo del trafico Obtención de datos: Tuvimos reuniones con la IMM que con muy buena disposición nos aporto datos relativos a la posición de los ómnibus? Con los datos realizamos un modelo vehicular de la ciudad [poner el mapa] que nos da una noción básica sobre el trafico en ciertos puntos de la ciudad, y la velocidad promedio de circulación. este modelo luego fue validado con revelamiento in-situ de diferentes puntos.

Tener en cuenta los cambios en el corredor: Buscar información publica sobre las luces- no hay- tomar medidas 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. [http://www.elobservador.com.uy/noticia/2 sincronizara-400-semaforos-para-agilizar-el-transito-capitalino/] buscar info sobre densidad - no hay- contar Cantidad de luces por semáforo Realizar nuevas mediciones de tiempos de luces Densidad del trafico actual (se puede conseguir o tomar mediciones) bondis: frecencias de bondis de donde sacamos esta info? autos: .py genera aleatorio

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 netconvert) 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

2.3. Herramientas

2.3.1. Open Street Map:

http://www.openstreetmap.org/about Es un proyecto en donde una comunidad crea mapas libres y editables. Cuenta con cerca de 1.8 millones de usuarios y mas de 20.000 que editaron algo en el ultimo mes [http://osmstats.neis-one.org/] por lo que es muy activa. Se utilizan datos de GPS móviles, fotografías satelitales y otras fuentes libres para crear los mapas y editarlos.

2.3.2. SUMO (Simulation of Urban MObility)

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

2.3.3. 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: http://sumo.dlr.de/wiki/Simulation/Output 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

2.3.4. NetConvert

Utilizado para la generación de la red a partir de un mapa. Por ejemplo podemos transformar un mapa de openStreetMap a archivo XML de red que sumo reconoce.

2.3.5. DUaRouter

Genera recorridos de vehículos basado en dinámicas definidas por el usuario.

2.4. Resumen

Capítulo 3

Presentación y trabajos relacionados

3.1. Presentación

3.2. Formulación matemática

3.3. Por que usar algoritmo genéticos?

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.

3.4. 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 métodos 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, ademas 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 utilizo 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 fija, obteniendo la solución propuesta mejores resultados en todos los casos evaluados

Este mismo grupo realizo 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 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

- 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 simulo tanto un caso estándar como casos de alta congestión de trafico, las comparaciones se hacen respecto a las distintas funciones de fitness y los distintos escenarios planteados logrando buenos resultados.

- 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 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 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 mas 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 mas 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 inteligente ya que indica que los atascos del trafico provocan no solo perdidas económicas sino también contaminación ambiental.

  Para ello propone utilizar un algoritmo inteligente que tomando en cuenta el estado de congestión de las rutas sugiere al usuario cual es la ruta mas 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 Open Street Map, cuenta con 8 entradas y 8 salidas , para la simulación utiliza SUMO. Los vehículos modelados son : turismo, monovolumen, furgoneta, camión donde se varia la longitud, velocidad y probabilidad que entre en la red de trafico.

  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 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 hasta con 800 vehículos,se concluye que al aumentar la cantidad de vehículos (mas 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, el cruzamiento toma 80 de información de un padre y 20 del otro. La función de fitness es el largo de las colas generadas. 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 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 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 mas 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. Ademas un algoritmo genético híbrido busca diferentes contantes que serán usadas en los arboles de decisión, permitiendo una especialización en las diferentes geometría y flujo de trafico

  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 trafico. 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 cuando 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 mas 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 trafico tanto en el corto como el largo plazo a través de la optimización de las señales de trafico 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 picos). Para solucionar esto proponen un sistema auto-adaptable que toma la información del trafico 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 mas fases mas 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 trafico. Este modelo es relativamente pequeño una intersección con 4 brazos, cada uno con 3 lineas 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 esta basado 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 mas 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 trafico con 9 intersecciones con semáforos en la ciudad de Chicago(Usa), contando con parking, rutas de ómnibus y paradas así como con vehículos. Se toman valores reales en horas pico AM y PM, comprobando que las colas que se generan en la simulación coinciden con la realidad. Usa el programa TRANSYT-7F[] (Que permite visualizar mapas y contiene optimización de varios algoritmos genéticos. Se probaron 12 estrategias distintas en 7F y la mejor fue simulada en CORSIM[] 100 veces. Se midió el tiempo de demora en la red y el largo de las colas producidas La performance de la red aumentó considerablemente usando este método

3.5. Resumen

Existen variedad de soluciones propuestas al problema de la sincronización de semáforos y todas logran buenos resultados en mayor o menor medida.

Como se aprecia analizando el estado del arte nuestra implementación es mucho más compleja que los tr

Capítulo 4

Estrategia de resolución

4.1. Framework

Malva [http://themalvaproject.github.io/] surge como un fork del proyecto Mallba [http://neo.lcc.uma.es/mallba/easy-mallba/index.html] . Propone la actualización, mejora y desarrollarlo como un proyecto de código abierto colaborativo. Su objetivo es proveer de varios esqueletos de heuristicas 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 un template genérico 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 dle problema particular. Los esqueletos provistos o sea los algoritmos soportados son Algoritmos genéticos y CHC.

4.2. Algoritmo Genético

El algoritmo que vamos a utilizar es el algoritmo simple, este sigue la propuesta de Goldberg []. El algoritmo en Malva se llama NewGA, 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.

4.2.1. Representación

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.

4.2.2. Codificación

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:

4.2.3. Inicialización

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).

4.2.4. Función 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.

4.2.5. Operadores

Selección

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.

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.

4.2.6. 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

4.3. Multiobjetivo

4.4. Algoritmo Paralelo

Agregar diagrama de maestro esclavo

4.5. Resumen

Capítulo 5

Análisis Experimental

En esta sección describimos los distintos escenarios que vamos a probar y los resultados obtenidos en cada uno de ellos así como comparativas que consideramos relevantes.

5.1. Descripción de escenarios

5.1.1. Caso base

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

5.1.2. Escenario Inicial

En este caso ejecutamos nuestro algoritmo evolutivo sobre el caso base para obtener una nueva sincronización de semáforos optimizada que repercutirá en la calidad del trafico.

5.1.3. Escenario Alternativo

Luego de analizar aquellos puntos que creemos atentan contra el buen funcionamiento del Corredor, agregamos algunas modificaciones al escenario base para intentar mejorarlo. Estos son limitar el numero de cruces en los que se puede doblar a la izquierda,

5.2. 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 paralelos 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.

5.3. Ajuste de parámetros de algoritmos

Buscamos la mejor configuración inicial de los parámetros realizando pruebas experimentales con diferentes combinaciones. Estos son: el tamaño de la población, probabilidad de mutación, probabilidad de cruzamiento, etc. Para esto se realizaron 20 ejecuciones independientes para el algoritmo secuencial inicial. El criterio de parada se eligió por el numero de generación.

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

La mejor configuración obtenida fue: Población:120, mutación:0.01 , cruzamiento: 1

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

5.3.1. Tiempo de simulación

Cada simulación tiene una duración fijada en base a ajustes previos que realizamos para que al menos 80 Esto nos permite mantener constante el tiempo de ejecución total del algoritmo y saber cuanto demorara su ejecución teniendo en cuenta la población y generaciones configuradas. De esta forma se logra una mayor confianza a la hora de comparar los resultados.

5.4. Resultados

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

5.4.1. Resultado simulación caso base

5.4.2. Resultado Escenario Inicial (Algoritmo Secuencial)

5.4.3. Resultado Escenario Alternativo (Algoritmo Secuencial)

5.4.4. Resultado de ejecución en paralelo.

5.4.5. 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 y luego al realizar los test T-student [] resulto tener menos de 0,0001 lo que se considera una diferencia que estadísticamente es significativa. Esto confirma algo que resulta 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. Comparación Algoritmo Secuencial vs Algoritmo Paralelo

Speedup se define como ...

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 resolver problemas de este tipo, obteniendo resultados muy buenos.

El enfoque multiobjetivo aun 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 este que abordamos. Y demuestran que son muy útiles en acelerar el procesamiento.

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.