Sistema dinámico y adaptativo para el control del tráfico de una intersección de calles: modelación y simulación de un sistema multi-agente

Tomás Fernandez Battolla[1], Santiago Fuentes[1], José Ignacio Illi[1], Joaquín Nacht[1], Mariana Falco[1], Gastón Pezzuchi[1], Gabriela Robiolo[1]

[1] Universidad Austral, Facultad de Ingeniería, LIDTUA (CIC), Pilar, Buenos Aires, Argentina

{tomas.fernandez, santiago.fuentes, jose.illi, joaquin.nacht}@ing.austral.edu.ar, {mfalco, grobiolo}@austral.edu.ar, gpezzuchi@gmail.com

Resumen. Es frecuente la pérdida de tiempo de los conductores y los embotellamientos de tráfico en las ciudades, por una configuración no óptima de los semáforos. Diversos dominios han intentado comprender el fenómeno e identificar las causas en pos de obtener una solución apropiada. El presente trabajo introduce un algoritmo cuyo fin es reducir el tiempo de espera de los conductores en una intersección. Para lo cual, la validación fue llevada a cabo en tres escenarios posibles, definidos por medio de la variación de la frecuencia de los autos. El entorno de simulación se implementó en NetLogo, lo que permitió estudiar el impacto del cambio de luces a través de la simulación de semáforos de tiempos fijos vs semáforos con control inteligente mediante agentes. Hemos obtenido una reducción máxima del 45% en el tiempo de espera. Finalmente, presentaremos las conclusiones y el trabajo futuro.

Palabras clave: Sistemas Multi-Agentes, Programación Orientada a Agentes, Tráfico, NetLogo.

1 Introducción

Las rutas y calles hacen posible la circulación y el recorrido de las ciudades, pero desafortunadamente el tráfico y particularmente la congestión del mismo conlleva a que los conductores sufran pérdidas de tiempo para llegar de un lugar a otro, por el tiempo que tardan en transitar sobre los caminos, sumado a los tiempos de espera por semáforos. Diversas disciplinas han estudiado dicha problemática en pos de comprender el fenómeno [1, 2, 3], y lograr una solución viable [4, 5, 6]. Algunos autores han encontrado que los conductores atascados en condiciones de alta congestión presentan niveles elevados de estrés [7], lo que puede llevarlos a modificar su comportamiento mientras manejan su vehículo. En consecuencia, es importante cambiar la forma en que las personas conducen, y encontrar mejores soluciones para lidiar con el tráfico, como semáforos inteligentes o incluso automóviles que se comunican entre sí.

El rápido crecimiento de las ciudades, la localización de las instituciones educativas y la diversidad de puestos de trabajo generan un aumento en el flujo diario del tráfico, lo que conlleva a un incremento en el número de vehículos y medios de transporte en

las calles en particular y en las ciudades en general. Si bien diversos dominios han abordado el problema de la congestión de tráfico [8, 9,10,11,12] consideramos que la implementación de un enfoque simple, que implemente técnicas de modelado y simulación basados en agentes, con un nivel de dificultad bajo-medio de implementación es capaz de ser efectivo y viable para su aplicación en la vida real. Existen un gran número de herramientas y plataformas como NetLogo [13] y Repast, y un elevado número de aplicaciones, donde los usos más comunes tienen lugar en simulaciones sociales y problemas de optimización como el comportamiento humano, la simulación urbana, la gestión del tráfico, entre otros [14].

En este contexto, el objetivo que persigue el presente artículo es introducir un nuevo algoritmo, en el contexto de una orientación a agentes, que disminuye el tiempo de espera de los vehículos, evitando que estos esperen un tiempo indeterminado o excesivamente largo; con la meta final de reducir la congestión de tráfico en una intersección de dos calles - doble mano y doble carril. Consecuentemente, en NetLogo se ha modelado y simulado la intersección con dos agentes: el agente auto y el agente intersección, donde el primero modela el comportamiento de los automóviles, mientras que el segundo, controla el flujo de autos. Vale destacar que solo se han modelado autos, sin peatones.

Por un lado, el agente Auto podrá avanzar, frenar, cambiar de carril y doblar a la derecha. Avanzará si no tiene otro vehículo adelante y se detendrá cuando el vehículo precedente se encuentre detenido o el semáforo esté en rojo al momento de llegar a la intersección. De la misma manera, podrá cambiar de carril y cuando llegue a la intersección podrá mantener su dirección o doblar a la derecha. Por el otro, el agente intersección tendrá dos estrategias de funcionamiento: tiempos fijos o tiempos variables, siendo la primera por tiempos determinados previamente, donde el semáforo estará en verde o rojo por un tiempo predeterminado. Los tiempos variables son dinámicos, por lo que la estrategia cambiará a partir de los datos sensados por el agente intersección. Así, tendrá en cuenta la cantidad de autos que están esperando, el tiempo que estuvieron esperando y el tiempo desde la última vez que estuvo en verde cada dirección.

Teniendo como base lo anterior, se han planteado tres escenarios donde varían la frecuencia de generación de autos en pos de poder efectivizar una comparación de las dos estrategias en cada uno de los escenarios para evaluar las mejoras obtenidas por el agente inteligente. El trabajo se estructura de la siguiente forma: en la sección 2, se contextualiza un breve estado de arte de sistemas basados en agentes aplicados al tráfico. La sección 3 explica el entorno de simulación e implementación del algoritmo, describiendo la herramienta elegida NetLogo, los agentes y sus funcionalidades, el ambiente y la interfaz de usuario, y la lógica y el código del algoritmo. La sección 4, busca validar el algoritmo implementado a partir de la comparación de las estrategias en los tres escenarios. Finalmente, la sección 5 presenta las conclusiones y los lineamientos del trabajo futuro.

2 Estado de arte

Uno de los primeros sistemas de control de tráfico fueron aquellos con planes de tiempos de intersección fijos y señalizados, donde las longitudes del ciclo variaban entre los 35 y los 120 segundos [15]. Tiempo después surgieron, los semáforos con control dinámico que utilizan sensores para detectar distintos tipos de vehículos y usuarios no motorizados como peatones; y el control coordinado lo que posibilita que los conductores encuentren una onda verde: una cadena de luces verdes en los semáforos, fenómeno conocido como progresión [16]. Finalmente, el control de tráfico adaptativo responde a una estrategia en la que el tiempo de la señal del semáforo cambia o se adapta según la demanda del tráfico en tiempo real.

Existen varias simulaciones desarrolladas con la plataforma NetLogo de sistemas dinámicos y adaptativos de control de tráfico basados en agentes, para una o varias intersecciones. Para el caso de una intersección, es viable mencionar a [17,18]. En el primero de ellos, Bui et al. reformulan los modelos de competencia económica de Cournot y Stackelberg para el desarrollo del algoritmo inteligente del semáforo, para vehículos con y sin prioridad. En este contexto, muestran una mejora en el tiempo de espera para dos condiciones, el flujo de vehículo uniforme y no uniforme en las dos direcciones, comparando el algoritmo propuesto versus uno basado en el método Round-Robin, con semáforos de tiempos fijos y el otro usando un sensor de red.

En el segundo ejemplo [18], desarrollan en el contexto de Internet de las Cosas, el modelado de la comunicación entre los agentes, para mejorar el flujo de tráfico en una intersección en tiempo real. Los vehículos se pueden identificar en función de su ID (por ejemplo, matrícula del vehículo), a partir de la cual también puede comunicarse con la infraestructura. Cada vehículo está equipado con el sistema de posicionamiento para determinar su ubicación. En el caso de los peatones, al usar dispositivos de conexión (por ejemplo, teléfonos inteligentes), pueden comunicarse con el sistema. La simulación mide el rendimiento de los algoritmos de control de tráfico al aumentar la densidad de los agentes que llegan a la intersección, considerando una distribución aleatoria para cada dirección de la intersección. Se compara la eficiencia del algoritmo propuesto versus un algoritmo de programación de Round-Robin para controlar el semáforo. Los resultados muestran que el tiempo de espera del vehículo obviamente aumenta cuando la densidad de los vehículos aumenta. Sin embargo, usando el algoritmo propuesto la variación no es tan significativa, dado que la prioridad de los mensajes de solicitud del vehículo está dada por un modelo FIFO, lo que permite que el vehículo pase la intersección lo antes posible. Concluyen que el algoritmo propuesto reduce el tiempo espera de los agentes.

Para el caso de múltiples intersección, en el contexto de complejas redes de tránsito desarrollaron un enfoque multi-agente utilizando clustering difusos para administrar el flujo del tráfico en condiciones determinadas, usando NetLogo para testear y evaluar el enfoque planteado. Concluyen que para una grilla de intersecciones, el tiempo medio de retardo se puede reducir en un 42,76% en comparación con la señal de tráfico de secuencia fija convencional [19].

Otros autores presentan en [20] un modelo adaptativo que modela una ciudad de 19x19 calles, comparando dos enfoques diferentes: a) en cada intersección se cuenta el número de automóviles que se aproximan a la luz roja, independientemente de su estado o velocidad; b) la duración del próximo ciclo verde de cada luz es directamente proporcional a la cantidad de automóviles que cruzaron la intersección en el último ciclo verde. Encuentran mejoras del segundo enfoque con respecto al primero, para la velocidad promedio de los autos, el número promedio de los autos parados y el tiempo medio de espera de los autos, para bajas densidades de autos y una mejora menor para altas densidades de autos. Concluyen que el segundo es más fácil de implementar debido a que es posible usar bandas de goma o detectores de bucle enterrados para la detección de autos.

Por último, otro artículo [21] presenta una simulación de la coordinación dinámica del sistema de semáforos urbanos, utilizando los datos de tráfico disponibles localmente y el estado del tráfico de las intersecciones vecinas, con el fin de mejorar tránsito urbano en la ciudad de Colima, México. Muestra la simulación de tres intersección (500 mts de distancia entre cada intersección) en una avenida. Se define y usa un algoritmo inteligente para el control adaptativo con capacidades de aprendizaje que permite un tráfico más fluido, reduciendo el tiempo de espera promedio y el tamaño medio de cola de espera por intersección. Compara la situación real con la propuesta, en escenarios de baja y alta afluencia de autos. Concluyen que el tiempo de espera para la dirección norte-sur se reduce un 16%, para el sur-norte el 59%, para el este-oeste el 57% y para el este-oeste hasta el 33% con respecto a la situación actual, para tráfico normal y un 29% para horas picos.

Los ejemplos citados demuestran que es posible reducir el tiempo de espera en una o varias intersecciones, aspecto que motiva presentar nuestros resultados, considerando que aún quedan tantas intersecciones o arterias en las ciudades en general y particularmente en Buenos Aires, donde es posible agilizar el tráfico.

3 Entorno de modelado, simulación e implementación del sistema

3.1 Modelado basado en Agentes mediante NetLogo

El modelado basado en agentes cuenta con un gran número de plataformas, que pueden dividirse en dos categorías. La primera define los modelos utilizando lenguajes de programación como Java o C++, donde las más conocidas son Swam y Repast; mientras que la segunda categoría utiliza lenguajes dedicados y las plataformas de código abierto más conocidas son NetLogo y GAMA [22]. NetLogo [13] creado en 1999 por U. Wilensky, es un entorno de modelado programable que permite simular fenómenos naturales y sociales, y es viable dar instrucciones a cientos o miles de agentes que operan independientemente lo que posibilita la exploración de la relación el comportamiento a nivel micro (individuos) y los patrones a nivel macro (interacciones). Es lo suficientemente simple para los estudiantes y profesores, pero lo suficientemente

avanzado como para servir como una poderosa herramienta para los investigadores en muchos campos. En NetLogo todo es un agente, desde cada cuadrado que conforma el plano donde se visualiza la simulación, hasta quien se encarga de pintar las calles, los semáforos y los autos. Además, ofrece una función <go> que ejecuta todo lo que esté dentro de ella a cada tick, considerando que un tick es una medida de tiempo que puede ser modificada. Se ejecuta en la JVM, por lo que funciona en todas las plataformas principales como Mac, Windows y Linux. Además, se ejecuta como una aplicación de escritorio y la operación de línea de comando también es compatible [23].

3.2 Descripción de los Agentes modelados

Los agentes implementados son los que se observan en las Tablas 1 y 2, donde se describen los agentes por sus atributos, por la información que sensa del ambiente y por su comportamiento.

Tabla 1. Descripción del Agente Auto

Descripción	**Agente Auto**
Atributos	• Velocidad: velocidad actual del auto • Paciencia: el nivel de tolerancia del conductor en la espera • Dirección: hacia dónde va el auto. • Carril actual. • Coordenadas actuales. • Girar (Sí o no): indica si el auto va a girar en la intersección. • Tiempo de espera en el semáforo. • Aceleración • Desaceleración
Información sensada del ambiente	• Autos cercanos: utilizando un ángulo de visión de 15 grados y un radio que varía dependiendo de la velocidad del auto. • Velocidad máxima de la calle • Color de la luz del semáforo • Autos en carril adyacente • Posición dentro del ambiente
Comportamiento	• Acelerar: lo hace si no tiene ningún auto en su espectro de visión yendo a una velocidad menor a la suya, si no supera la velocidad y si no se encuentra en el límite previo al semáforo, mientras este está en rojo. • Frenar: lo hace si tiene ningún auto en su espectro de visión yendo a una velocidad menor a la suya y si se encuentra en el límite previo al semáforo, mientras este está en rojo. • Doblar a la derecha: lo hace sí se encuentra en el carril derecho al momento de llegar al área crítica de la intersección con una probabilidad definida por una variable. • Cambiar de carril: lo hace sí la paciencia del conductor llega a 0 (está va disminuyendo a medida que el tiempo de espera aumenta) y si no hay ningún auto en su posible futura posición.

Tabla 2. Descripción del Agente Intersección

Descripción	Agente Intersección
Atributos	• Luces de los semáforos: estado actual de los semáforos • Inteligencia (Sí o no): define el algoritmo a utilizar para el manejo de luces de los semáforos, o sea una estrategia de tiempo dinámica o estática. • Tiempo mínimo de verde • Tiempo desde que se realizó el último cambio de luz
Información sensada del ambiente	• Cantidad de autos detrás de los límites de la intersección en todas las direcciones. • Tiempos de espera de los autos
Comportamiento	Cambiar luz de semáforos: • Tiempos fijos o sin inteligencia: lo realiza por tiempos definidos, el semáforo está en verde por un tiempo determinado y después cambia a rojo. • Tiempos variables o con inteligencia: lo realiza a partir de los datos censados del ambiente. A partir de un algoritmo que cuenta cuántos autos están esperando del otro lado y de cuánto tiempo llevan esperando decide si cambiar o seguir en el estado actual.

3.3 Algoritmo del semáforo inteligente: lógica y código

El algoritmo del semáforo inteligente desarrollado, cuyo código puede leerse en la Fig.1, coordina dos semáforos, uno para cada dirección de la intersección de las calles; considera la cantidad de autos que están esperando en ambas direcciones o ejes, siendo el eje la dirección del flujo de vehículos que tiene el semáforo en luz verde y la dirección del flujo de autos que tienen luz roja.

Por lo tanto, cuenta con: (a) la cantidad de autos en el eje x antes de cruzar la intersección, o sea los autos en dirección este-oeste, y le aplica la sustracción con los que ya cruzaron la intersección y, (b) la cantidad de autos en el eje y antes de cruzar la intersección, o sea, los autos en dirección norte-sur y le aplica la diferencia con los que ya cruzaron la intersección.

Se consideran los pesos de cada una de las direcciones, valorando la cantidad de autos y el tiempo de espera que provoca el semáforo en rojo. Al cambiar a verde, siempre va a cumplir un tiempo mínimo definido y a continuación, el algoritmo cal-culará los pesos de ambas direcciones. Vale destacar que la luz verde corresponde al semáforo con mayor peso y los pesos del eje i, se calculan según (1):

$$Pi = Ai + Tsri * 0.001 \quad (1)$$

donde:

- A_i, los autos previos a la intersección en el eje i (autos esperando antes de cruzar la intersección)

- T_{sri}, el tiempo de semáforo en rojo en el eje i (tiempo desde que el semáforo cambió a rojo hasta el instante actual)

```
to change-lights-intelligent

  ;Cars heading east and west that did't reach the critic zone yet
  set x-waiting-cars count cars with [(direction = 1 or direction = 3) and not pass-intersection? and out-of-intersection]
  ;Cars heading north and south that did't reach the critic zone yet
  set y-waiting-cars count cars with [(direction = 0 or direction = 2) and not pass-intersection? and out-of-intersection]

  ;Sets the direction of the red light to the correct direction
  ask one-of lights with [color = red or color = green + 0.1] [
    set red-direction light-direction
  ]

  ;Ticks since last light change
  let red-time (ticks - ticks-at-last-change)
  ;If is the lights are red to north and south
  if-else (red-direction = 0 or red-direction = 2) [
    ;set the weight as the cars waiting plus the time the light has been red times 0.001(empiric factor)
    set y-waiting-cars (y-waiting-cars + red-time * 0.001)
  ] [
    set x-waiting-cars (x-waiting-cars + red-time * 0.001)
  ]

  ;If there are more cars waiting on the x axis than on the y axis and it is red on the x axis
  ;and its been in red for more time than the bottom limit it changes the light to yellow and the to green
  if(x-waiting-cars > y-waiting-cars and (red-direction != 0 and red-direction != 2) and elapsed? green-length )[
    change-to-yellow
  ]
  if(y-waiting-cars > x-waiting-cars and (red-direction != 1 and red-direction != 3) and elapsed? green-length )[
    change-to-yellow
  ]
  ;If the yellow time is over and there are any lights in yellow it changes them to red
  if elapsed? yellow-length  and any? lights with [ color = green + 0.1 ] [change-to-red]

end
```

Fig. 1. Algoritmo de control implementado en NetLogo.

Vale destacar que Tsri es multiplicado por un factor cuyo valor fue obtenido por medio de pruebas experimentales, por lo que para el eje que está en verde, Tsri va a ser igual a 0. Por ejemplo, para un tiempo mínimo de luz verde de 200 ticks (unidad de tiempo en NetLogo), el semáforo correspondiente al eje x (calle horizontal) acaba de cambiar su luz a verde, y el eje y (calle vertical) tiene el semáforo en rojo, consecuentemente la luz del semáforo permanecerá en verde el tiempo definido en un principio (200 ticks). Pasado el tiempo mínimo comienza a evaluarse la fórmula del peso en ambos ejes:

$$Px = Ax + Tsrx * 0.001 \quad (2)$$

$$Py = Ay + Tsry * 0.001 \quad (3)$$

En el instante que Tsry sea igual a 1000 ticks, Ax sea igual a 10 y Ay sea igual a 5, Px será mayor que Py (Px >Py) , o sea 10>6, por lo que el semáforo del eje x va a continuar en rojo; mientras que en el instante que Tsry sea igual a 1100 ticks, Ax sea igual a 7 y Ay sea igual a 6, Px será menor que Py (Px <Py) , o sea 7<7.1, entonces el semáforo del eje x va a cambiar a rojo y del eje y a verde.

El tiempo medio de espera es la sumatoria de todas las esperas de los autos y se calcula de la siguiente manera:

$$average\text{-}waiting\text{-}time = \frac{(average\text{-}waiting\text{-}time \times dead\text{-}cars) + waiting\text{-}time}{dead\text{-}cars + 1} \quad (4)$$

donde dead-cars, son los autos que ya salieron del sistema, y waiting-time, es el tiempo que el auto saliente esperó, tal que:

$$waiting\text{-}time = totalTicks - first\text{-}stop\text{-}time \quad (5)$$

Este tiempo abarca desde que el auto frenó antes de cruzar la intersección hasta que cruzó la misma. A su vez, siendo:

- ticks, la cantidad total de ticks (unidad de tiempo utilizada) transcurridos
- first-stop-time, la cantidad de ticks transcurridos cuando el auto frenó.

3.4 Ambiente e interfaz de usuario

Se han considerado los siguientes supuestos, que están aplicados y pueden visualizarse en la Fig. 2: (a) el único tipo de vehículo modelado es el auto, (b) las calles que conforman la intersección poseen una velocidad máxima que los autos respetan, (c) las calles que conforman la intersección son doble mano, (d) cuando un auto llega a la intersección lo único que puede hacer es cruzarla o girar a la derecha, (e) para doblar a la derecha los autos deben estar en el carril derecho de la calle, (f) los sensores de la intersección no fallan en ningún momento, (g) no hay disturbios externos al sistema, y (h) el peatón ha sido excluido del modelo. Las variables mostradas en la Figura 2 se pueden observar también en la Tabla 3.

Recordando la definición de Wooldridge [24], un agente se encuentra en un ambiente e interactúa con él y con otros agentes, con los cuales se comunica, coopera y negocia. En este caso y debido a su modelización en NetLogo, está constituido por agentes pasivos o patches que se encargan inicialmente de decorar el entorno. También, está compuesto por dos ejes: el horizontal y el vertical, donde cada uno contiene una calle compuesta por dos manos y dos carriles en cada dirección (un total de 4 carriles). Las líneas amarillas que se visualizan en la Figura 2 son las encargadas de dividir las manos de una misma calle mientras que las blancas punteadas dividen los carriles. El verde alrededor de la calle es meramente decorativo y no posee ninguna interacción con los agentes.

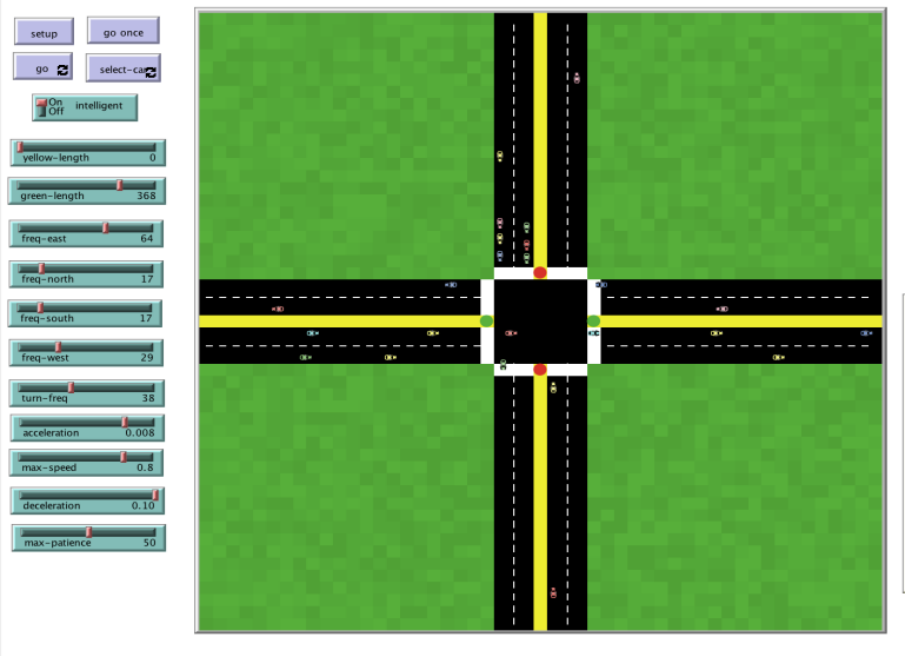


Fig. 2. Modelo y GUI en NetLogo.

Ahora bien, el área crítica de la intersección está delimitada por las líneas blancas gruesas del centro de la imagen, que indican hasta dónde pueden avanzar los autos si el semáforo de su eje se encuentra en rojo, representados como puntos circulares que están sobre las líneas blancas y en el punto medio de cada una de ellas. Finalmente, cada línea blanca tiene un semáforo asociado, siendo los semáforos del mismo eje siempre del mismo color.

La Fig. 2 muestra el diseño de la interfaz de usuario, que permite asignar valores a las variables y hacer el setup inicial. Como puede observarse, en la parte superior izquierda se encuentran los botones básicos para una simulación de NetLogo: <setup> que realiza la configuración inicial de la aplicación, <go> que hace correr repetidamente el programa (secuencia de ticks), y finalmente, <*go-once*> que hace que pase un tick únicamente. Luego, <*select car*> permite ver el estado de un auto al hacerle click, y es una de las formas de realizar el *debugging* de la aplicación.

El switch de on/off permite seleccionar la estrategia de comportamiento del agente Intersección, es decir: semáforo de tiempo fijo o con inteligencia. Los sliders de <*yellow-length*> y <*green-length*> cambian la duración de las luces de los semáforos del agente intersección. En el caso de que la estrategia sea de tiempo fijo, el <green-length> representa el tiempo fijo en el que el semáforo irá cambiando de luz, mientras que si la estrategia es de tiempo dinámico, el <*green-length*> representa el tiempo mínimo de verde. Los sliders de frecuencia indican con qué probabilidad se crean los agentes autos desde cada punto de generación, y se basa en una distribución aleatoria en cada tick. Finalmente, los últimos sliders indican la frecuencia de giro, la acelera-

ción, la velocidad máxima, la desaceleración y la paciencia máxima. Al modificarlos, se cambian esos atributos en los autos.

4 Validación del sistema

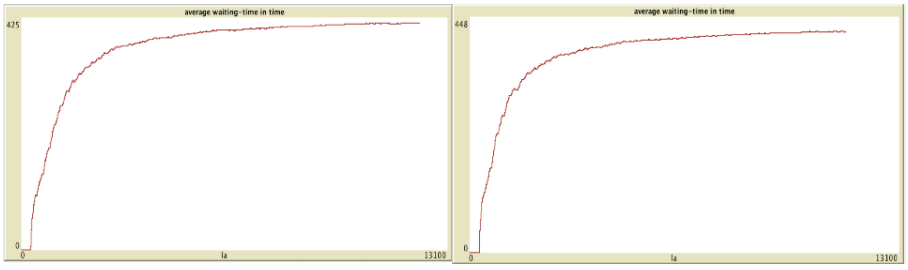
Con el fin explícito de validar el sistema, se han planteado tres escenarios en los cuales se varía la frecuencia de generación de autos: igual, ligeramente diferente y diferente, comparando un semáforo de cambio de colores en tiempos fijos versus un semáforo inteligente o de tiempo dinámico. En la Tabla 3 se pueden observar los valores iniciales de cada escenario.

Tabla 3. Valores iniciales de cada escenario.

Variables	Igual frecuencia	Frecuencia ligeramente diferente	Frecuencia diferente
yellow-length	0	0	0
green-length	194	194	194
freq-east	64	64	64
freq-north	17	17	6
freq-south	17	17	7
freq-west	17	35	64
turn-freq	38	38	38
acceleration	0.008	0.008	0.008
max-speed	0.8	0.8	0.8
deceleration	0.06	0.06	0.06
max-patience	50	50	50

4.1 Primer scenario: igual frecuencia

En este caso, los ejes tienen igual frecuencia de generación de autos por lo que ambos tendrán aproximadamente la misma cantidad de autos.



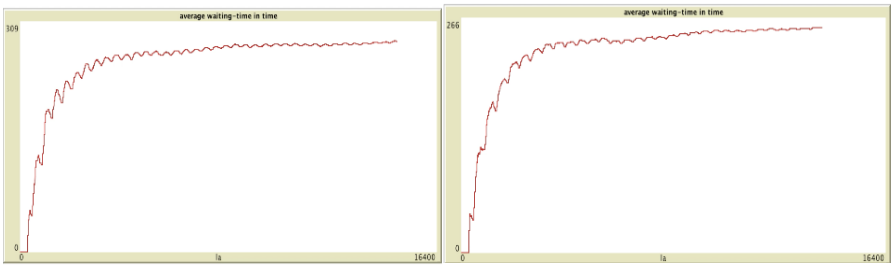
a) tiempo fijo b) tiempo dinámico

Fig. 3. Tiempo promedio de espera con igual frecuencia de generación autos para semáforos de tiempo de espera fijo y tiempo dinámico.

La Fig. 3. a) muestra la variación de la curva de tiempo promedio de espera y su estabilización en un valor de 415, y la suma de las frecuencias de los ejes (horizontal: este y oeste; vertical: norte y sur) tienen una relación de 1 a 1. La Fig. 3.b) muestra la variación de la curva de tiempo promedio de espera y su estabilización en un valor 419. Podemos concluir que en este escenario no se observan diferencias significativas entre los tiempos promedios de espera del sistema para cada una de las estrategias, debido a la igualdad de la frecuencia de generación autos, o sea a un flujo similar en ambas direcciones, lo cual implica que el algoritmo no produce una mejora en el tiempo de espera, puesto que el beneficiar una dirección perjudica a la otra.

4.2 Segundo escenario: frecuencia ligeramente diferente.

En este segundo escenario, los ejes tienen una frecuencia ligeramente distinta de generación de autos, por lo que el eje horizontal tendrá mayor caudal de autos ya que sus frecuencias suman 99, versus los 34 sumados por el eje vertical. La relación entre ambos es aproximadamente de 3 a 1. Se observan diferencias entre las dos estrategias, con una mejora del tiempo de espera del 9.9%. La Fig. 4.a) visualiza cómo, a medida que avanza el tiempo, la curva de tiempo promedio de espera se va estabilizando en el valor 282, y la relación entre la frecuencia del eje y con respecto al eje x es de 3 a 1. Luego, la Fig. 4.b) muestra que el valor del tiempo promedio de espera se estabiliza alrededor del valor 254. Conceptualmente, se observan diferencias entre ambas estrategias, con una mejora del tiempo de espera del 9.9%.



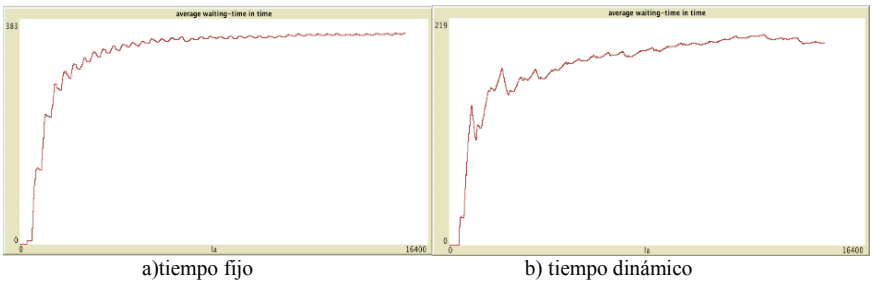
a) tiempo fijo b) tiempo dinámico

Fig. 4. Tiempo promedio de espera con igual frecuencia de generación autos ligeramente diferente para semáforos de tiempo de espera fijo y tiempo dinámico.

4.3 Tercer escenario: frecuencia diferente.

Finalmente, los ejes tienen frecuencia muy distinta de generación de autos, por lo que el eje horizontal tendrá mucho mayor caudal de autos ya que sus frecuencias suman

128, versus los 13 sumados por el eje vertical. La relación entre ambos es aproximadamente de 10 a 1.



a)tiempo fijo b) tiempo dinámico

Fig. 5. Tiempo promedio de espera con igual frecuencia diferente de generación autos ligeramente diferente para semáforos de tiempo de espera fijo y tiempo dinámico.

La Fig. 5.a) muestra cómo a medida que avanza el tiempo la curva de tiempo promedio de espera se va estabilizando en el valor 359, y la relación de las frecuencias es aproximadamente de 10 a 1. Mientras que la parte b) muestra que el valor de tiempo promedio de espera se estabiliza alrededor de 195. En este escenario se observan diferencias relevantes entre los tiempos promedios de espera comparando ambas estrategias. Se obtuvo una mejora del 45%.

4.4 Discusión de Resultados

El algoritmo inteligente presentado logra una mejora en el caso de que exista una diferencia entre la cantidad de autos que circulan en una y otra dirección. Cuanto mayor sea la diferencia, mayor será la mejora obtenida en el tiempo de espera. Particularmente, a una relación de 3 a 1, se obtuvo una mejora del 9.9% y a una relación de 10 a 1 del 45%. En el caso de una cantidad de autos similares en ambas direcciones, el algoritmo no tiene capacidad de mejora, y presenta un comportamiento similar para tiempos fijos o dinámicos. Este resultado es esperable, porque se pretende optimizar la demora en ambas direcciones. El algoritmo busca disminuir el tiempo de espera de los autos en ambas direcciones, por lo tanto, la dirección que tenga mayor caudal de autos se va a ver beneficiada con mayor tiempo de luz verde. Al mismo tiempo se evita el estancamiento de autos en la dirección de menor flujo, asegurando un tiempo mínimo de permanencia de la luz verde. Vale destacar que el coeficiente de corrección del tiempo de la luz roja Tsri, al aplicarlo a un caso real se deberá ajustar convenientemente.

5 Conclusiones

El tráfico es una realidad que afecta a muchas personas, y existen evidencias de la disminución del tiempo de espera de los conductores si se implementan sistemas dinámicos y adaptativos de control de semáforos. El enfoque orientado a agentes tiene

características esenciales que facilitan su modelización y simulación. Existen diversas aplicaciones, sistemas y simulaciones que posibilitan que el área se mantenga activa, debido a que el incremento en la demanda de la sociedad en pos de una mejora no solo es en la infraestructura sino también en los sistemas de gestión.

Considerando que a medida que aumenta la cantidad de vehículos circulando en un área se vuelve mayor el tiempo de espera que producen los semáforos, el presente trabajo aporta una nueva evidencia sobre la posibilidad de disminuir el tiempo de espera de los autos en una intersección de calles, cuando existe una diferencia entre la cantidad de autos que circulan en una y otra dirección, en una intersección compuesta por calles doble mano y doble carril, y autos que sólo pueden avanzar o girar a la derecha. Concretamente se obtuvo una mejora del 45% en el tiempo promedio de espera de los autos, para una relación de 10 a 1 entre los flujos de los autos en las dos direcciones.

El resultado presentado es un ensayo inicial, susceptible de ser mejorado, al incorporar otros agentes, como peatones, ambulancias, colectivos, camiones, motos, animales, agentes de tráfico o situaciones de accidentes. Además, el ambiente de implementación utilizado (NetLogo) brinda un entorno que posibilita escalar este caso a una red compleja de tráfico, con más de una intersección. Es conveniente destacar que el ambiente de simulación tiene una curva de aprendizaje baja para personas que tengan conocimientos básicos de programación, pero tiene la dificultad de no favorecer el trabajo en grupo porque no es posible acceder al código, más de una persona en el mismo momento.

También destacamos, que si bien es posible escalar la aplicación, a medida que es más grande es más difícil encontrar los errores de programación puesto que no se puede realizar una descomposición modular. Considerando que la validación realizada fue llevada cabo con datos que no representan una situación real relevada, como trabajo futuro se prevé efectivizar su validación en un caso real, con tomas de datos de frecuencias de autos verdaderas, en distintos momentos del día.

Agradecimientos. El presente proyecto se ha realizado con el apoyo de la Universidad Austral.

Referencias

1. Long, J., Gao, Z., Ren, H., & Lian, A.: Urban traffic congestion propagation and bottleneck identification. Science in China Series F: Information Sciences, 51(7), 948-964 (2008)
2. Rao, A. M., & Rao, K. R.: Measuring Urban Traffic Congestion-A Review. International Journal for Traffic & Transport Engineering, 2(4), 286-305 (2012).
3. Tan, F., Wu, J., Xia, Y., & Chi, K. T.: Traffic congestion in interconnected complex networks. Physical Review E, 89(6), 062813 (2014)
4. Lindley, J. A.: Urban freeway congestion: quantification of the problem and effectiveness of potential solutions. ITE journal, 57(1), pp. 27-32 (1987)
5. Bauza, R., & Gozálvez, J.: Traffic congestion detection in large-scale scenarios using vehicle-to-vehicle communications. Journal of Network and Computer Applications, 36(5), 1295-1307 (2013)

6. Lopez-Garcia, P., Onieva, E., Osaba, E., Masegosa, A. D., & Perallos, A.: A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy. IEEE Transactions on Intelligent Transportation Systems, 17(2), 557-569 (2016)
7. Hennessy, D. A., & Wiesenthal, D. L.: Traffic congestion, driver stress, and driver aggression. Aggressive behavior, 25(6), 409-423 (1999)
8. Putha, R., Quadrifoglio, L., & Zechman, E.: Comparing ant colony optimization and genetic algorithm approaches for solving traffic signal coordination under oversaturation conditions. Computer-Aided Civil and Infrastructure Engineering, 27(1), 14-28 (2012)
9. Teodorović, D., & Dell'Orco, M.: Mitigating traffic congestion: solving the ride-matching problem by bee colony optimization. Transportation Planning and Technology, 31(2), 135-152 (2008)
10. Bazzan, A. L., & Klügl, F.: A review on agent-based technology for traffic and transportation. The Knowledge Engineering Review, 29(3), 375-403 (2014)
11. Chen, B., & Cheng, H. H.: A review of the applications of agent technology in traffic and transportation systems. IEEE Transactions on Intelligent Transportation Systems, 11(2), 485-497(2010)
12. Ehlert, P. A., & Rothkrantz, L. J.: Microscopic traffic simulation with reactive driving agents. In Intelligent Transportation Systems. Proceedings. 2001 IEEE. pp. 860-865(2001)
13. Tisue, S., & Wilensky, U.: NetLogo: Design and implementation of a multi-agent modeling environment. In Proceedings of agent (Vol. 2004, pp. 7-9) (2004)
14. Allan, R. J.: Survey of agent based modelling and simulation tools (pp. 1362-0207). Science & Technology Facilities Council, Technical report (2010)
15. Smith, S. F., Barlow, G. J., Xie, X. F., & Rubinstein, Z. B.: Smart Urban Signal Networks: Initial Application of the SURTRAC Adaptive Traffic Signal Control System. Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling (ICAPS) (2013)
16. Nagatani, T.: Vehicular traffic through a sequence of green-wave lights. Physica A: Statistical Mechanics and its Applications, 380, 503-511 (2007)
17. Bui, K. H. N., Jung, J. E., & Camacho, D.: Game theoretic approach on Real-time decision making for IoT-based traffic light control. Concurrency and Computation: Practice and Experience, 29(11) (2017)
18. Bui, K. H. N., Camacho, D., & Jung, J. E.: Real-time traffic flow management based on inter-object communication: a case study at intersection. Mobile Networks and Applications, 22(4), 613-624, Springer (2017)
19. Daneshfar, F., RavanJamJah, J., Mansoori, F., Bevrani, H., & Azami, B. Z.: Adaptive fuzzy urban traffic flow control using a cooperative multi-agent system based on two stage fuzzy clustering. In Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th (pp. 1-5). IEEE (2009)
20. Burguillo-Rial, J. C., Rodríguez-Hernández, P. S., Montenegro, E. C., & Castiñeira, F. G.: History-based self-organizing traffic lights. Computing and Informatics, 28(2), 157-168 (2012)
21. Guerrero-Ibanez, A., Contreras-Castillo, J., Buenrostro, R., Marti, A. B., & Muñoz, A. R.: A policy-based multi-agent management approach for intelligent traffic-light control. In Intelligent Vehicles Symposium (IV), 2010 IEEE (pp. 694-699). IEEE (2010)
22. Banos, A., Lang, C., & Marilleau, N.: Agent-based spatial simulation with NetLogo (Vol. 1). Iste Press Ltd. y Elsevier (2015)
23. Wilensky, U., & Rand, W.: An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo. MIT Press (2015)
24. Wooldridge, M.: An introduction to multiagent systems. John Wiley & Sons (2009)