Particle Swarm Optimization Aplicado a la Programación de los Ciclos de Semáforos en Bahía Blanca

José M. García-Nieto[a*], Enrique Alba[a] y Ana C. Olivera[b]

[a]Dept. de Lenguajes y Ciencias de la Computación, University of Málaga,
ETSI Informática, Campus de Teatinos, Málaga - 29071, España

[b]Departamento de Ciencias e Ingeniería de la Computación,
Universidad Nacional del Sur,
Av. Alem 1253 - 8000, Bahía Blanca, Argentina

Resumen En este trabajo, proponemos el uso de un algoritmo de optimización mediante cúmulos de partículas (PSO, por las siglas en inglés), el cual, utilizando el simulador de tráfico SUMO para su entrenamiento, es capaz de obtener programas de ciclos de semáforos optimizados para áreas urbanas extensas. Como caso de estudio, nos hemos centrado en la ciudad de Bahía Blanca, Argentina. Las planificaciones de ciclos resultantes, tras una serie de experimentos, obtienen reducciones significativas en términos de congestión de tráfico y tiempo medio de viaje.

Keywords: Ciclos de semáforo, Particle Swarm Optimization, SUMO.

1. Introducción

En la actualidad, el número de semáforos en las ciudades está creciendo y su planificación conjunta es cada vez más compleja debido a la gran cantidad de combinaciones de periodos de colores que se deben manejar. Esto hace necesario la utilización de sistemas automáticos inteligentes para la programación óptima de ciclos de semáforos. Recientemente, las iniciativas más prometedoras en esta línea están enfocadas en el uso de simuladores [1,2,3] y la programación eficiente de ciclos de luces en semáforos [4,5]. En este sentido, el uso de metaheurísticas [6,7], ha ido ganando en importancia demostrando su capacidad para la planificación de semáforos [2,8,9,10]. Sin embargo, el uso de dichos sistemas inteligentes se limita por lo general a la optimización de instancias académicas pequeñas, con una o dos intersecciones diseñados para áreas urbanas muy específicas, con escasos vehículos y semáforos. Además, los resultados reportados en estos trabajos no fueron comparados con otras técnicas ni con planificaciones generadas por personal experto.

* Dirección del autor. E-mail: {jnieto,eat}@lcc.uma.es, aco@cs.uns.edu.ar.

En este trabajo presentamos un algoritmo PSO (Particle Swarm Optimization) [11] como optimizador de programas de ciclos de semáforos. Para la evaluación de los programas de ciclos generados (codificados como vectores solución), se utilizó el simulador de tráfico microscópico SUMO (Simulator of Urban Mobility) [12]. Este simulador trabaja a modo de caja negra que tiene como entrada el programa de ciclos para una determinada instancia de área urbana y devuelve los valores de aptitud de dicho programa para la instancia evaluada. Para este trabajo se emplearon tres instancias localizadas en el centro urbano de Bahía Blanca. Los resultados y comparaciones con otras técnicas: Random Search (RANDOM) y Sumo Cycle Program Generator (SCPG) [12]; ponen de relieve las mejoras significativas obtenidas por nuestra propuesta en términos de flujo de tráfico y de tiempo medio de viaje.

El trabajo se organiza de la siguiente manera. En la Sección 2, se presenta el problema de la programación óptima de ciclos de luces en semáforos. La Sección 3 describe nuestra estrategia de optimización. Las Secciones 4 y 5 presentan los experimentos llevados a cabo y los subsecuentes análisis, respectivamente. Las conclusiones se detallan en la Sección 6.

2. Optimización de Programas de Ciclos en Semáforos

Los semáforos se sitúan en las intersecciones de las calles y controlan el flujo del tráfico mediante sus programas de estados de colores y periodos de duración. Todos los semáforos en una misma intersección están gobernados por un programa común. En nuestro modelo trabajamos exclusivamente con combinaciones válidas de estados de colores para cada intersección.

En este contexto, nuestro principal objetivo consiste en encontrar **programas de ciclos optimizados** (OPCS) para todos los semáforos situados en una determinada área urbana. Nos referimos a los programas de ciclos (o lógicas de tráfico) como el periodo de tiempo -en segundos- en el que un conjunto de semáforos (en una intersección o cruce) permanecen con sus estados de luces/colores.

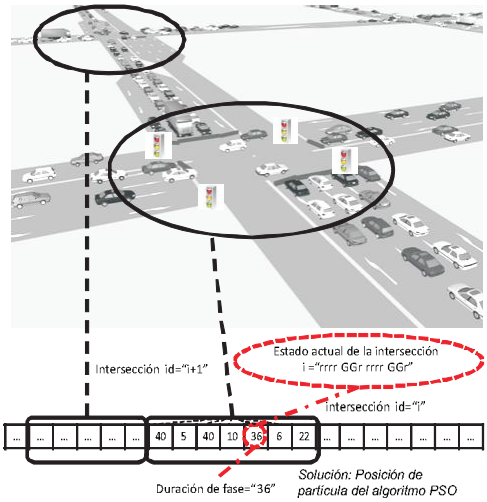


Figura 1. Programa de ciclos en un cruce

2.1. Codificación del OPCS

Para nuestra estrategia hemos codificado el OPCS mediante un vector de números enteros (positivos) siguiendo la estructura de SUMO para la programación de ciclos, a través de la cual, cada elemento del vector (variable) representa una duración de fase de los semáforos implicados en una determinada intersección (Figura 1).

2.2. Función de Fitness

Para evaluar cada programación generada por nuestro algoritmo se ha formulado la siguiente función de aptitud (a minimizar):

$$aptitud = \frac{TV + TE + (ND \cdot TS)}{V^2 + P} \tag{1}$$

El objetivo principal (Ecuación 1) consiste en maximizar el número de vehículos que alcanzan sus destinos (V) mientras se minimiza el tiempo medio de viaje de todos los vehículos (TV) durante el tiempo de simulación (TS). El número de vehículos que llegan a sus destinos se eleva al cuadrado (V^2) para hacerlo prioritario sobre los demás términos y factores. El número de vehículos que no alcanzan sus destinos durante la simulación (ND) debe ser minimizado. El tiempo medio de viaje se refiere a la agregación de los tiempos de viaje de todos los vehículos que alcanzan sus destinos durante el tiempo de simulación. Por el contrario, los vehículos que no completen sus viajes serán considerados con tiempo de viaje igual al tiempo de simulación. Otro importante término considerado en esta función es el estado en el que los semáforos están en un preciso momento, ya que éste influencia el tiempo que cada vehículo debe parar y esperar (TE), con el consecuente retardo en su propio tiempo de viaje. Finalmente, una proporción bien balanceada de colores en las duraciones de fase de los estados debería promocionar aquellos estados con más semáforos en verde situados en vías o calles con un gran número de vehículos circulando, y los semáforos en rojo situados en calles con un bajo volumen de tráfico. La proporción de colores en cada fase (pf) de todas las intersecciones int se formula mediante la Ecuación 2.

$$P = \sum_{k=0}^{int} \sum_{j=0}^{pf} d_{k,j} \cdot \left(\frac{v_{k,j}}{r_{k,j}} \right), \tag{2}$$

Donde $v_{k,j}$ es el número de semáforos en verde y $r_{k,j}$ es el número de semáforos en rojo en el estado j (con duración de fase $d_{k,j}$) y en la intersección k. Para evitar la división por cero se uso $r_{k,j}$ igual a 1 como valor mínimo.

3. Estrategia de Optimización

Nuestra estrategia se compone por dos partes principales: la optimización con PSO y la simulación con SUMO. El algoritmo de PSO [11] es una metaheurística poblacional inspirada en el comportamiento social de las bandadas de aves y las agrupaciones de individuos en general. En PSO, cada solución potencial al problema se codifica mediante la *posición* de una partícula y a la población de partículas se le llama *cúmulo* o *enjambre (swarm)*. Para el desarrollo de nuestro PSO hemos seguido las especificaciones del estándar de 2007 [13]. En este algoritmo, cada posición de partícula x^i se actualiza en cada iteración g mediante la Ecuación 3, donde el término v^i_{g+1} es la velocidad de la partícula (Ecuación 4).

$$x_{g+1}^i = x_g^i + v_{g+1}^i \tag{3}$$

$$v_{g+1}^i = w \cdot v_g^i + U[0,\varphi_1] \cdot (p_g^i - x_g^i) + U[0,\varphi_2] \cdot (b_g^n - x_g^i) \tag{4}$$

En esta fórmula, p_g^i es la mejor solución personal que la partícula i ha encontrado durante su proceso, b_g^n es la mejor partícula en un vecindario de n partículas (conocida como *el mejor social*) aleatoriamente seleccionado (distribución uniforme) del cúmulo y w es el factor de inercia de la partícula (que controla el balance entre exploración-explotación). Por último, φ_1 y φ_2 son los coeficientes de aceleración, los cuales controlan el efecto relativo de los mejores personal y social de la partícula, mientras que $U[0,\varphi_k]$ es un valor aleatorio uniforme en el intervalo $[0,\varphi_k],\ k \in 1,2$. Este último se genera de nuevo para cada componente en el vector de velocidad y para cada iteración. Debido a que el OPCS se codifica mediante vectores de números naturales (representando ciclos de duración de fase), hemos utilizado el método de cuantificación (*quantisation*) provisto por el estándar PSO 2007 [13]. El método de cuantificación se aplica a cada nueva partícula (Ecuación 3) y transforma las variables continuas a discretas. Básicamente consiste en un cuantificador uniforme de Mid-Thread como se especifica en la Ecuación 5. El paso de cuantificación está inicializado como $\Delta = 1$.

$$Q(x) = \Delta \cdot \lfloor x/\Delta + 0{,}5 \rfloor \tag{5}$$

Algoritmo 1 Estándar PSO 2007 para OCP

```
1: inicializaCúmulo()
2: while g < maxIteraciones do
3:    for partícula x_g^i do
4:       b_g^n=mejorVecino(x_g^i, n)
5:       v_{g+1}^i=actualizaVel(w, v_g^i, x_g^i, φ1, p_g, φ2, b_g^n)    //Eq. 4
6:       x_{g+1}^i=Q(actualizaPos(x_g^i, v_{g+1}^i))   //Eqs. 3 y 5
7:       evalúa(x_{g+1}^i)   //SUMO Simulación y Eq. 1
8:       p_{g+1}^i=actualiza(p_g^i)
9:    end for
10: end while
```

El pseudocódigo mostrado en Algoritmo 1 describe el Estándar PSO 2007 para OCPS. El mismo comienza con la inicialización de las partículas del cúmulo (Línea 1). Los elementos correspondientes de cada posición de partícula (solución) son inicializados con valores aleatorios representando las duraciones de fase. Estos valores son generados en el intervalo de tiempos $[5,60] \in Z^+$. Para un número máximo de iteraciones, cada partícula se mueve a través del espacio de búsqueda mediante la actualización de su velocidad y posición (Líneas 4, 5, y 6), se evalúa (Línea 7) y su mejor posición personal se actualiza también (Línea 8). Finalmente, el algoritmo devuelve como resultado la mejor partícula encontrada durante todo el proceso de optimización. La simulación se utiliza para asignar

un valor cuantitativo de aptitud a las soluciones codificadas en las partículas, para un determinado escenario urbano. Así, cuando el algoritmo PSO genera una nueva solución con un nuevo programa de ciclos, se inicia SUMO para simular la instancia con calles, direcciones, obstáculos, semáforos, vehículos, velocidades, rutas, etc., respecto al nuevo programa de ciclos. Tras la simulación, SUMO devuelve los valores necesarios para computar la función de aptitud (Ecuación 1).

4. Experimentos

Para este trabajo nos hemos basado en un escenario generado a partir de información real en mapas digitales. Este escenario cubre un área de aproximadamente 0,42 km^2 y está físicamente localizado en la ciudad de Bahía Blanca, Argentina. La ilustración de la Figura 2 muestra el área seleccionada para Bahía Blanca tratada con sus correspondientes capturas de Google Maps, OpenStreetMap y SUMO, describiendo el proceso de generación de los escenarios urbanos.

El área urbana seleccionada se encuentra enmarcada por la *Plaza Rivadavia*, esta plaza se encuentra situada en el centro comercial de la ciudad de Bahía Blanca (Figura 2). Se seleccionaron 53 intersecciones a su alrededor. Las calles forman una conformación regular de grilla de bloques, característico de las ciudades de Argentina. Excepto Avda. Colón y Alem la mayoría de las calles poseen un solo sentido con el contrario en las calles aledañas. Mejorar el flujo de tráfico en esta zona eliminaría problemas de congestión debidos principalmente a lo angosto de sus calles y la imposibilidad de realizar una profunda reestructuración en las mismas. El Cuadro 1 contiene la combinación de lógicas de tráfico y vehículos utilizada para cada instancia en el escenario de la Plaza Rivadavia. En total son 9 instancias con 20, 30 y 40 lógicas de tráfico que consideran el estado actual de la red de tráfico y la generación de semáforos nuevos en intersecciones que actualmente no los poseen. Cada uno de los vehículos recorre su propia ruta desde el origen hasta su destino circulando con una velocidad máxima de 50 km/h. Las rutas fueron generadas a priori aleatoriamente. El tiempo de simulación es de 500 segundos para cada instancia. Cuando un vehículo abandona el recorrido del escenario, ya no volverá a aparecer de nuevo durante la simulación. Para la ejecución del algoritmo hemos utilizado la implementación en C++ de PSO que provee la biblioteca MALLBA [6]. La fase de simulación se realiza mediante el simulador SUMO en su versión

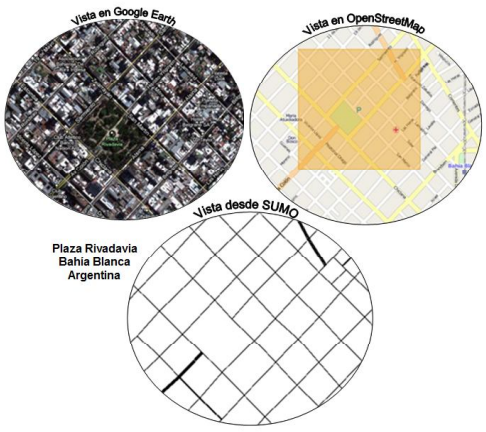


Figura 2. Plaza Rivadavia vista con Google Maps, Openstreetmap y SUMO

0.12.0 para Linux. Los experimentos se realizaron en las computadoras de los laboratorios del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, España. La mayoría de estas máquinas están equipadas con procesadores dual core, con 1GB de RAM y sistema operativo Linux Devian. Todas ellas operan bajo la plataforma middelware Condor [14], que actúa como un planificador de tareas distribuido (cada tarea se encarga de una ejecución independiente de PSO).

Cuadro 1. Instancias

Núm. de Lógicas	Núm. de semáforos	Núm de vehículos
20	88	100
		300
		500
30	136	100
		300
		500
40	176	100
		300
		500

Cuadro 2. Parámetros de Simulación y de PSO

Fase	Parámetro	Valor
Simulación	Tiempo de Simulación (pasos)	500 s
	Area de Simulación	0,42 km^2
	Numero de Vehículos	100/300/500
	Velocidad de Vehículo	0-50 km/h
	N. de Intersecciones	20/30/40
PSO	Max. N. de Evaluaciones	30.000
	Tamaño del Cúmulo	100
	Tamaño de Partícula	88/136/176
	Coeficiente Local (φ_1)	2,0
	Coeficiente Social (φ_2)	2,0
	Máximo de Inercia (w_{max})	0,5
	Mínimo de Inercia (w_{min})	0,1

Para cada instancia se realizaron 30 ejecuciones independientes de nuestro PSO. El tamaño del cúmulo es de 100 partículas, realizando 300 iteraciones de optimización (resultando de esta manera unas 30.000 evaluaciones) por ejecución. Como ya comentamos en la sección anterior, el tamaño de la partícula depende directamente del número de semáforos de la instancia. Los parámetros están resumidos en el Cuadro 2, siendo éstos el resultado de ejecuciones preliminares de nuestra estrategia de optimización con el escenario de Bahía Blanca, 30 lógicas de tráfico y 300 vehículos en circulación. Respecto a los parámetros específicos de PSO, se utilizaron aquellos recomendados en el estudio sobre la convergencia de este algoritmo en [15] y que se pueden ver en el Cuadro 2.

Algoritmo 2 Pseudocódigo de RANDOM

```
1: inicializaSolución(x)
2: i ← 0
3: while i < maxEvaluaciones do
4:    genera(x_i)    //nueva solución
5:    if f(x) ≥ f(x_i) then
6:       x ← x_i
7:    end if
8:    i ← i + 1
9: end while
```

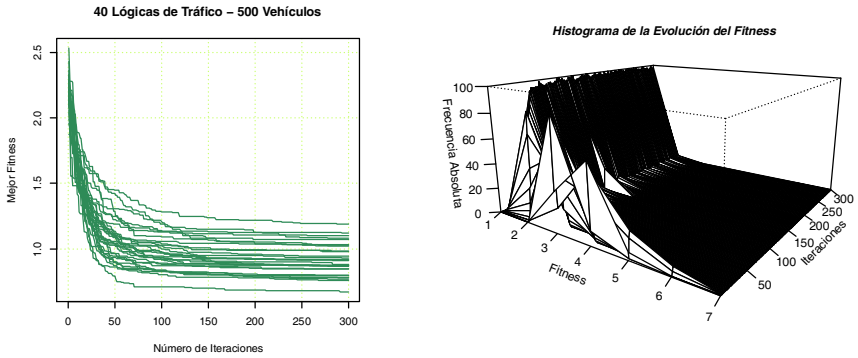
Con el fin de poder establecer comparaciones, hemos implementado además un algoritmo de Random Search, también en el ámbito de la biblioteca MALLBA

(Algoritmo 2). El número máximo de evaluaciones a realizar por RANDOM fue también iniciado a 30.000. Por otra parte, SUMO provee de un algoritmo determinista para la generación de programas de ciclos (SCPG) que consiste básicamente en asignar a las duraciones de fase de los semáforos nuevos valores en el rango [6,31], de acuerdo a tres factores diferentes: a) La proporción de estados en verde de las fases, b) el número de vías de entrada en las intersecciones, c) el tiempo de frenado de los vehículos cuando se acercan a los semáforos. La información completa sobre el funcionamiento del algoritmo SCPG puede ser encontrada en [12].

4.1. Resultados y Comparaciones

En primer lugar, en esta sección analizamos el comportamiento interno de nuestro PSO en la programación óptima de ciclos de semáforos.

Gráficamente, en la Figura 3(a) podemos observar las trazas de progreso de las 30 ejecuciones independientes de PSO en la planificación de la instancia de Bahía Blanca con 500 vehículos en circulación y 40 lógicas de tráfico. Todas las soluciones finales resultaron con cierta dispersión en cuanto a la calidad del fitness y en cuanto a su composición. En términos de convergencia y robustez, éstas son características deseables ya que así podemos ofrecer a los expertos en el área un conjunto variado de programas de ciclos desde un estado inicial de la optimización.



(a) Trazas de progreso de PSO para 40 intersecciones y 500 vehículos

(b) Histograma de valores de aptitud del cúmulo completo para 30 lógicas de tráfico y 500 vehículos

Figura 3. Resultados sobre distintas instancias de Bahía Blanca

Respecto a cada ejecución independiente, la Figura 3(b) muestra un ejemplo representativo de traza donde podemos observar la frecuencia absoluta de la distribución de fitness del cúmulo completo. En concreto, se refiere a una ejecución de nuestro PSO en la resolución del escenario con 30 lógicas de tráfico y 500

vehículos. En esta gráfica, las partículas iniciales presentan valores de aptitud diversos y altos ($\simeq 7$), para ir convergiendo durante la segunda mitad del proceso de optimización hacia soluciones con valores de aptitud (≤ 1) similares y bajos. En esta ejecución específica, 475 vehículos alcanzaron su destino de los 500 (95 %) antes de los 500 segundos. Este comportamiento es similar para todas las ejecuciones independientes del PSO.

En cuanto a los resultados generales, el Cuadro 3 contiene la media de los valores de aptitud obtenidos (en las 30 ejecuciones independientes) por la estrategia propuesta para todas las instancias. Junto a éstos se presentan en el mismo cuadro los resultados referentes a los algoritmos RANDOM y SCPG. De este modo, podemos comprobar que PSO obtuvo los mejores resultados (en negrita). Hemos aplicado diferentes *t-tests* [16] a los valores numéricos y las distribuciones de los resultados. El nivel de confianza utilizado fue del 95 % (*p-value*=0,05), lo cual nos permite asegurar que dichas distribuciones son estadísticamente diferentes si los test resultan con un *p-value*<0,05. Efectivamente, el *t-test para muestras independientes* aplicado a las distribuciones de PSO y RANDOM (Cuadro 3) resultó con *p-values* $\ll$ 0,05, para las tres instancias: 20, 30 y 40. De manera similar, el *t-test para una muestra* aplicado a la media de la distribución de PSO contra el valor de SCPG también resultó con *p-values* $\ll$ 0,05. Por tanto, podemos destacar que PSO obtuvo resultados estadísticamente mejores que los otros dos algoritmos comparados: RANDOM (con búsqueda totalmente estocástica) y SCPG (determinista).

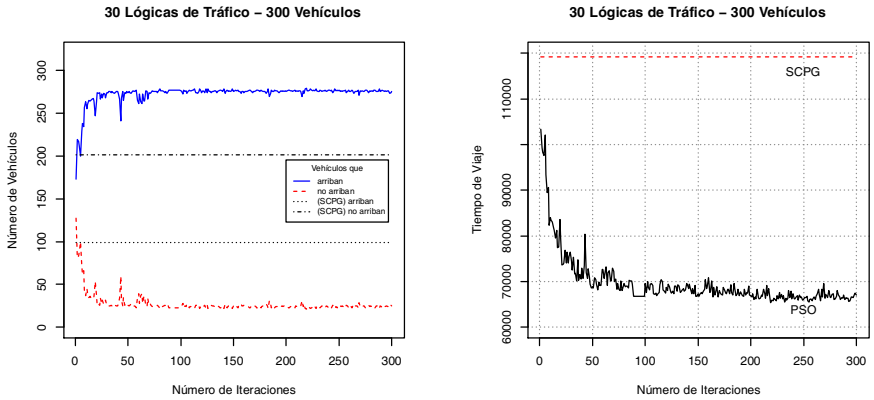
Cuadro 3. Valores medios de aptitutd obtenidos por PSO, RANDOM y SCPG para todas las intancias de Bahía Blanca. NTL es el número de lógicas de tráfico.

NTL	Numero de vehículos								
	100			300			500		
	PSO	RANDOM	SCPG	PSO	RANDOM	SCPG	PSO	RANDOM	SCPG
20	**1.64E+00**	2.91E+00	2.38E+00	**8.40E-01**	1.45E+00	9.24E-01	**7.93E-01**	1.51E+00	9.56E-01
30	**1.80E+00**	3.11E+00	2.45E+00	**9.09E-01**	1.65E+00	9.57E-01	**8.79E-01**	1.72E+00	9.89E-01
40	**1.79E+00**	3.08E+00	2.49E+00	**9.11E-01**	1.75E+00	9.76E-01	**8.96E-01**	1.74E+00	9.93E-01

5. Análisis de los Programas de Ciclos Resultantes

Para cara iteración de PSO y cada partícula del cúmulo, hemos tomado toda la información obtenida de cada simulación realizada. Esta información consiste en el número de vehículos que alcanzan su destino durante el tiempo de simulación, la duración media de sus viajes y el momento en el que abandonaron el escenario. De esta forma podemos distinguir la mejora progresiva en el flujo del tráfico con respecto a las soluciones iniciales.

En la Figura 4(a) se muestra la traza (ejecución de la mediana) respecto al número de vehículos que alcanzaron sus puntos de destino (curva continua superior) versus el número de vehículos que no alcanzaron sus destinos (curva discontinua inferior). Los resultados de SCPG están mostrados para la misma instancia mediante líneas rectas discontinuas. En términos de flujo de tráfico,



(a) Número de vehículos que llegaron (no llegaron)

(b) Tiempo medio de viaje

Figura 4. Análisis de los tiempos de viaje y vehículos que alcanzaron su destino

280 vehículos de los iniciales 300 (93.33 %) finalizaron satisfactoriamente sus trayectos. Más aún, en las soluciones finales de PSO, una media de 275 vehículos completaron sus viajes (media de 30 ejecuciones). Esto último contrasta con los 99 vehículos que completaron sus trayectos con el programa de ciclos de SCPG, resultado una mejora del 58.66 % de PSO sobre SCPG. La Figura 4(b) nos muestra la traza del tiempo medio de viaje empleado por los vehículos con los programas de ciclos generados por PSO durante todas las evaluaciones de una ejecución. En este caso, el tiempo de viaje es cada vez más corto a medida que el algoritmo alcanza su condición de parada.

6. Conclusiones

En este trabajo presentamos una estrategia de optimización basada en PSO para encontrar programas de ciclos de semáforos de manera automática. Los experimentos y análisis realizados con el sistema propuesto se presentan desde dos enfoques diferentes: el comportamiento de la técnica de optimización utilizada y la calidad de las soluciones generadas en el dominio de la planificación de semáforos. Nuestra estrategia muestra un comportamiento satisfactorio en escenarios de diseño realista. Para las tres instancias estudiadas, nuestro PSO obtiene resultados estadísticamente mejores que los otros dos algoritmos comparados: el generador de programas de ciclos de SUMO (SCPG) y el de Random Search. Además, las soluciones finales obtenidas por PSO mejoran tanto en el número de vehículos que llegan a sus destinos como en la duración de viaje.

Agradecimientos

Este trabajo está financiado por la Junta de Andalucía (CICE), mediante el proyecto P07-TIC-03044, el Ministerio Español de Ciencia e Innovación (MICINN) y FEDER

con los proyectos TIN2008-06491-C04-01 y TIN2011-28194; además del PICT 2011-0639 de la ANPCyT y el PGI 24/N026 SeCyT-UNS. José García-Nieto disfruta de una beca con código BES-2009-018767 de MICINN.

Referencias

1. J. McCrea y S. Moutari, "A hybrid macroscopic-based model for traffic flow in road networks," *Euro. Jour. of Oper. Res.*, vol. 207, no. 2, pp. 676-684, 2010.
2. J. Sánchez, M. Galán y E. Rubio, "Applying a traffic lights evolutionary optimization technique to a real case: "Las Ramblas"área in Santa Cruz de Tenerife," *IEEE Trans.s on Evol. Comp.*, vol. 12, no. 1, pp. 25-40, feb. 2008.
3. C. Karakuzu y O. Demirci, "Fuzzy logic based smart traffic light simulator design and hardware implementation," *App. Soft Comp.*, vol. 10, no. 1, pp. 66-73, 2010.
4. E. Brockfeld, R. Barlovic, A. Schadschneider y M. Schreckenberg, "Optimizing traffic lights in a cellular automaton model for city traffic," *Phys. Rev. E*, vol. 64, no. 5, pp. 56-132, 2001.
5. T. Nagatani, "Effect of speed fluctuations on a green-light path in a 2d traffic network controlled by signals," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 19, pp. 4105-4115, 2010.
6. E. Alba, G. Luque, J. García-Nieto, G. Ordonez y G. Leguizamón, "Mallba: a software library to design efficient optimisation algorithms," *Int. J. of Innovative Computing and Applications 2007 (IJICA)*, vol. 1, no. 1, pp. 74-85, 2007.
7. C. Blum y A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Computing Surveys (CSUR)*, vol. 35, no. 3, pp. 268-308, 2003.
8. J. García-Nieto, E. Alba y A. C. Olivera, "Swarm intelligence approach for the traffic light scheduling: Application to real urban áreas," *Engineering Applications Of Artificial Intelligence*, vol. 25, num. 2, pp. 274-283, 2012.
9. L. Peng, M. Wang, J. Du y G. Luo, "Isolation niches particle swarm optimization applied to traffic lights controlling," in *48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, dec. 2009, pp. 3318-3322.
10. F. Teklu, A. Sumalee y D. Watling, "A genetic algorithm approach for optimizing traffic control signals considering routing," *Computer-Aided Civil and Infrastructure Engineering*, vol. 22, pp. 31-43, 2007.
11. J. Kennedy y R. C. Eberhart, *Swarm Intelligence*, Morgan Kaufmann Publishers, San Francisco, 2001.
12. M. Behrisch,L. Bieker, J. Erdmann y D. Krajzewicz. "SUMO - Simulation of Urban MObility: An Overview SIMUL 2011," *The Third International Conference on Advances in System Simulation*, pp. 63-68, 2011.
13. M. Clerc et al., "Standard PSO 2011," Tech. Rep. [online] http://www.particleswarm.info/, Particle Swarm Central, January 2011.
14. D. Thain, T. Tannenbaum y M. Livny, "Distributed computing in practice: the condor experience.," *Concurrency - Practice and Experience*, vol. 17, no. 2-4, pp. 323-356, 2005.
15. M. Clerc y J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58-73, 2002.
16. D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, CHAPMAN & HALL/CRC STATISTICS, D. J. Sheskin, UK, 2003.