Algoritmo Memético para Calibrar Modelos de Micro-simulación de Flujo de Tráfico Vehicular CORSIM

Cristian David Arteaga Sánchez
Carlos Fabián Gaviria Molano

Director: Ing. Ember Ubeimar Martínez Flor
Co-Director: Ing. Alexander Paz Cruz

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Sistemas
Línea de Investigación de Sistemas Inteligentes
Popayán, febrero de 2015

TABLA DE CONTENIDO

INDICE DE FIGURAS

INDICE DE TABLAS

LISTADO DE ACRONIMOS

GA: Algoritmo memético instanciado mediante algoritmo genético sin optimizador local
GASA: Algoritmo memético instanciado mediante algoritmo genético y recocido simulado
GATS: Algoritmo memético instanciado mediante algoritmo genético y búsqueda tabú
SPSA: Algoritmo de Perturbación Simultanea y Aproximación Estocástica
FHWA: Federal Highway Administration
NDOT: Departamento de transporte de Nevada
NRMS: Media cuadrática normalizada

LISTADO DE ANEXOS

Anexo 1. Metodología de desarrollo del prototipo software
Anexo 2. Artículo de investigación y acuse de recibo en una revista especializada.
Anexo 3. Manual de referencia CORSIM. (En CD-ROM)

CAPITULO 1

1. INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

Un modelo de simulación es la representación de un sistema, usualmente su propósito es ayudar a entender o mejorar un sistema [1]. Los modelos de simulación de tráfico vehicular se valen de grandes y variados conjuntos de datos para poder representar con mayor exactitud diversos fenómenos, dichos datos son llamados parámetros [2]. En los modelos de simulación de flujo de tráfico vehicular CORSIM, de ahora en adelante modelos, se encuentran diversos grupos de parámetros llamados “Record Type”, estos conjuntos de datos representan diversas formas en las que un comportamiento se puede exhibir sobre un link(unión entre dos nodos los cuales representan intersecciones de una o más calles) durante una simulación [3], por ejemplo el record type 21 contiene los parámetros que describen la dirección del flujo vehicular (izquierda, derecha, seguir derecho, tomar una diagonal) que puedan ocurrir en una intersección del modelo [4], adicionalmente sobre esta intersección el record type 21 considera la restricción de los movimientos ya mencionados. La Figura 1 expone la distribución de datos en un modelo (los modelos contienen la información en archivos de texto plano, estos archivos tienen extensión .trf), la Figura 2 a manera de ejemplo muestra parte de los datos contenidos en un modelo, la Figura 3 muestra la representación gráfica de un modelo renderizado a partir de los datos del modelo.

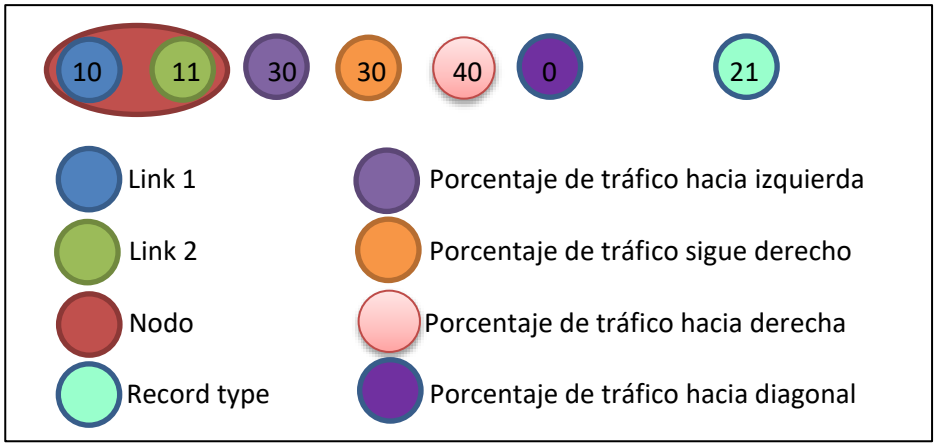
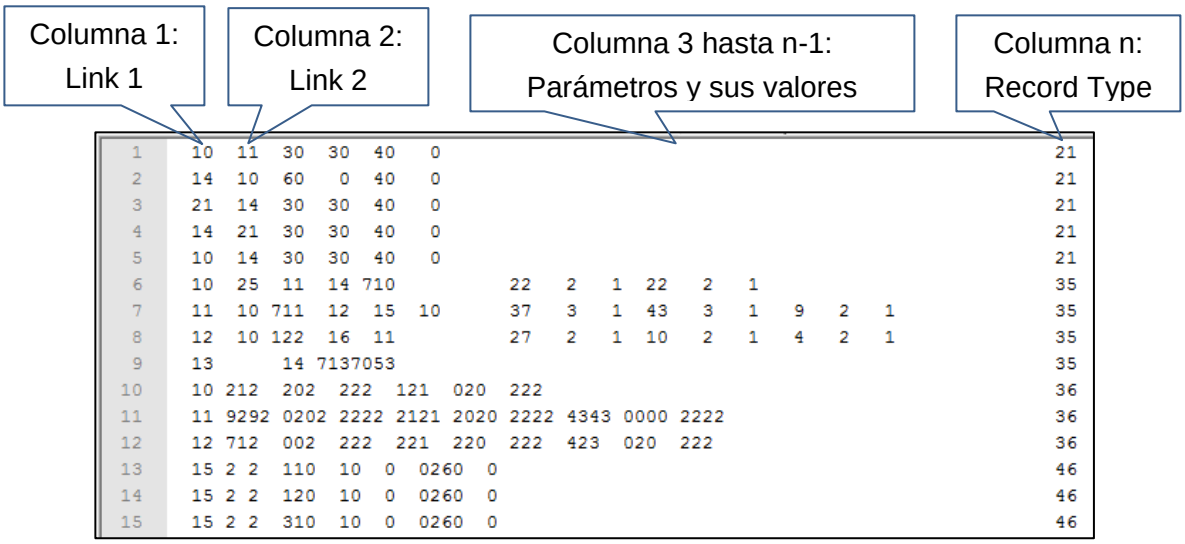


Figura 1. Explicación de comportamiento de modelo en texto plano

Los modelos de simulación de flujo de tráfico vehicular CORSIM, son procesados al ser simulados mediante el microsimulador CORSIM, los microsimuladores procesan los datos

del modelo tomando en cuenta el comportamiento de cada uno de los individuos (para CORSIM los individuos son los vehículos que se encuentran en el modelo), los datos de salida del microsimulador son descritos como la acumulación de estadísticas medidas durante la simulación y son de gran relevancia al realizar diversos estudios sobre los comportamientos de los modelos [5].




```
10  11  30  30  40   0                                                  21
14  10  60   0  40   0                                                  21
21  14  30  30  40   0                                                  21
14  21  30  30  40   0                                                  21
10  14  30  30  40   0                                                  21
10  25  11  14 710          22   2   1  22   2   1                      35
11  10 711  12  15  10      37   3   1  43   3   1   9   2   1          35
12  10 122  16  11          27   2   1  10   2   1   4   2   1          35
13      14 7137053                                                      35
10 212  202  222  121  020  222                                        36
11 9292 0202 2222 2121 2020 2222 4343 0000 2222                        36
12 712  002  222  221  220  222  423  020  222                        36
15 2 2  110  10  0  0260  0                                            46
15 2 2  120  10  0  0260  0                                            46
15 2 2  310  10  0  0260  0                                            46
```

Figura 2. Ejemplo de modelo en texto plano, archivo .trf

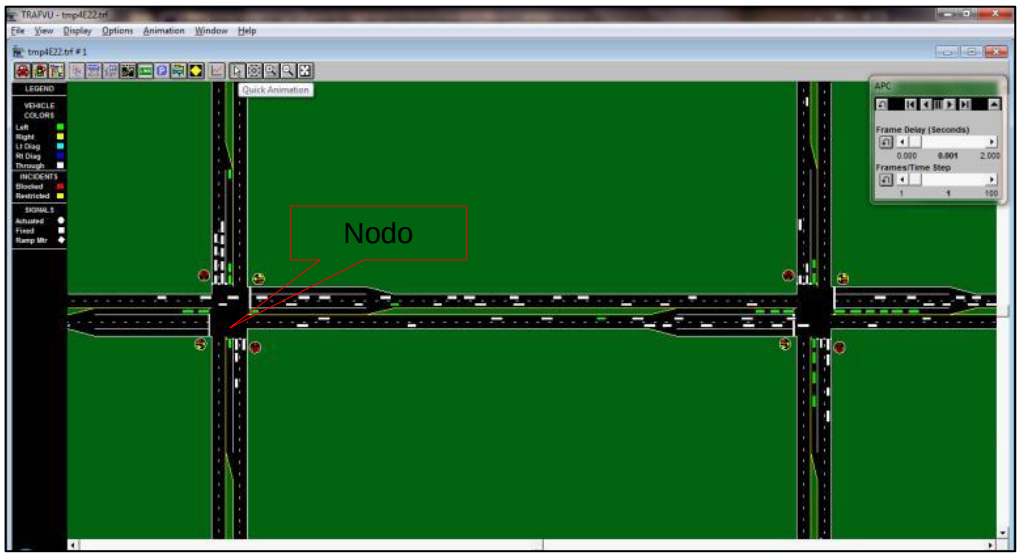


Figura 3. Representación gráfica de un modelo

El simulador CORSIM procesa los modelos a través de dos componentes, uno de estos componentes es NETSIM el cual se encarga de procesar los datos de las calles en el modelo, el otro es FRESIM encargado de procesar los datos de las avenidas en el modelo, estos componentes trabajan en conjunto cuando se realiza una simulación [6].

Dentro de los datos de salida se pueden encontrar diversas medidas, dentro de las medidas que se obtienen al procesar modelos CORSIM están las medidas de emisiones de gases, de volumen del flujo vehicular o de velocidad del flujo vehicular, estas medidas pueden ser usadas para realizar diversas investigaciones, por ejemplo se han realizado estudios [7] que sirven para ayudar a determinar que modificaciones son necesarias para mejorar zonas de embotellamientos o de alta accidentalidad. A partir de las medidas de emisión de gases se podrían realizar trabajos que permitan determinar el impacto medio ambiental sobre una determinada zona.

Generalmente los modelos de simulación intentan replicar comportamientos que se presentan en el mundo real. Determinar los valores de los parámetros que un modelo necesita para representar los comportamientos que se presentan en el mundo real es una tarea compleja ya que algunos de estos valores son difíciles o imposibles de extraer, por ejemplo el tratar de medir en cada vehículo del modelo el número de cambios de carril que realizó en un determinado tramo o medir el tiempo de frenado que requiere cada vehículo para detenerse son tareas difíciles. Si se tiene en cuenta que estos modelos tienen cientos de tipos de parámetros y además que cada modelo puede contener cientos o miles de vehículos se puede concluir que desde su creación un modelo no puede contener la información necesaria para representar fiablemente el comportamiento del mundo real [8].

La calibración de modelos se realiza con el fin de ajustar los valores de los parámetros del modelo [9]. Frecuentemente el proceso de calibrar modelos de simulación se realiza a través de modificaciones en los parámetros del modelo, buscando que los datos de salida de la simulación del modelo se asemejen en la mayor medida posible a datos obtenidos a través de sensores u observación del lugar en el que se presenta el fenómeno que se intenta replicar a través del modelo de simulación [10], estos datos son llamados datos de campo. De esta forma los modelos calibrados pueden representar de forma más fiable el comportamiento observado en campo pues los datos de salida o resultados de la simulación se acercan a los datos que se obtuvieron de las medias de campo. El proceso de calibración se describe en la Figura 4.

Para poder cuantificar y determinar la diferencia o cercanía numérica de los conjuntos de datos de salida y datos de campo, se hace necesario calcular un error aproximado entre los conjuntos, durante el presente trabajo para tal fin se utilizara la técnica llamada media cuadrática normalizada [11].

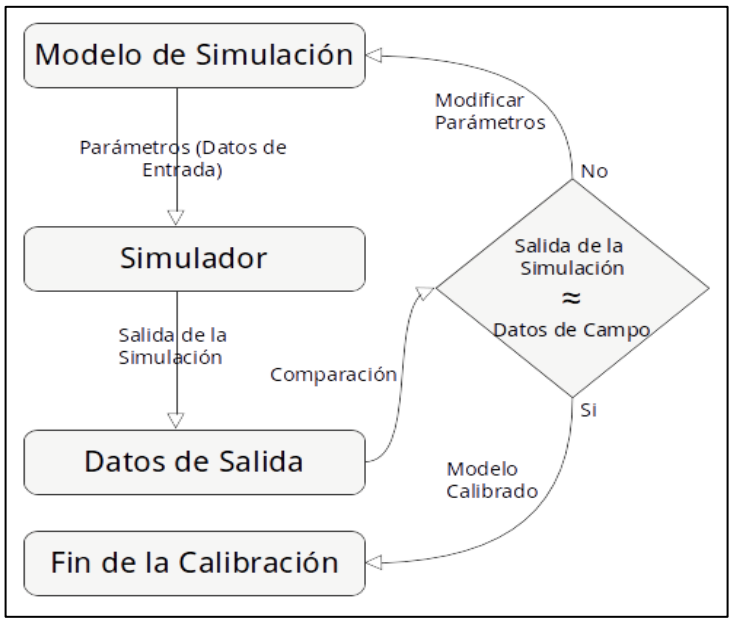


Figura 4. Proceso de calibración

El proceso de modificar los parámetros durante una calibración permite realizar la búsqueda del conjunto de valores necesarios para que un modelo se considere calibrado [12], uno de los problemas más complejos que se presenta al realizar estas modificaciones es que mientras se cambia el valor de un parámetro los demás se ven afectados, de esta forma al tener en cuenta el gran número de parámetros que llega a tener un modelo, la tarea de determinar el valor adecuado para cada parámetro es inviable, así la modificación de los parámetros debe realizarse en grupo, es decir todos los parámetros deben modificarse y posteriormente ser evaluados.

Durante el presente trabajo se tomarán en cuenta las especificaciones de la FHWA Federal Highway Administration [6] para determinar cuándo un modelo se considera calibrado. Durante el capítulo 5 del presente documento se explicaran más a fondo dichas especificaciones.

Principalmente al realizar una calibración se necesita encontrar la mejor solución (expresada como el conjunto de valores sobre los parámetros necesario para que un modelo se considere calibrado) en un espacio de posibles soluciones. Esta necesidad se puede suplir a través de la aplicación de una técnica llamada optimización [13]. La Figura 5 describe el proceso de optimización.

Durante el desarrollo del presente trabajo se plantea calibrar modelos a través de un algoritmo memético [14]. Dadas las características que le posibilitaron a este tipo de algoritmos haber encontrado soluciones precisas sobre grandes espacios de búsqueda en diversos problemas (se exponen en el estado del arte) que presentan características

similares al problema de calibración aquí expuesto se plantea la hipótesis de que la correcta instancia y parametrización de un algoritmo memético podría servir como algoritmo de optimización para calibrar un modelo.

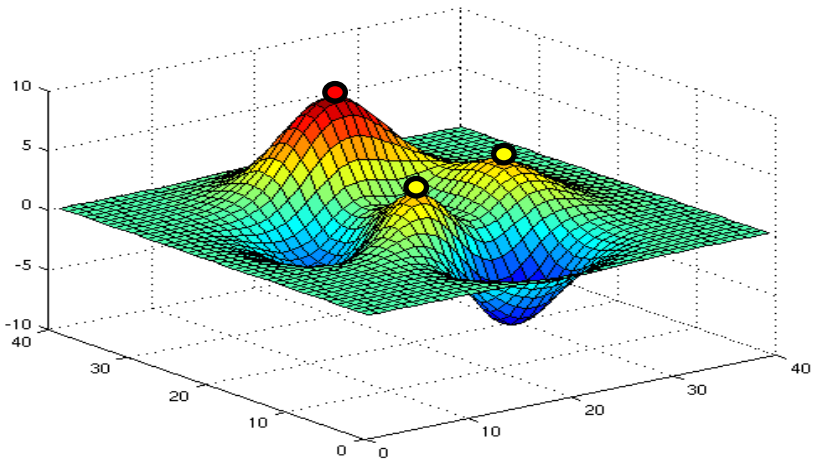
La definición del algoritmo memético será expuesta más a fondo en el marco teórico del presente documento, la argumentación de la escogencia del algoritmo y la escogencia de sus instancias será presentada durante la justificación y experimentación.

Para poder probar la mencionada hipótesis se realizarán experimentos con los que se verificará si un algoritmo memético puede o no calibrar modelos que partan de una solución distante de ser una solución óptima para el problema que presenta la calibración (con el fin de evitar procesos de sensibilización de parámetros o pre-procesamientos en el modelo), además el algoritmo debe poder realizar la calibración de todos los parámetros calibrables del modelo simultáneamente y tener en cuenta más de una medida.

El presente trabajo se presenta como la continuación de un trabajo realizado durante los años 2010 a 2012 llamado “Calibration of CORSIM models considering all model parameters simultaneously” [15] sobre este trabajo se realizará una comparativa a partir de los resultados obtenidos en la presente investigación. Los datos de campo y modelos con los que se trabajó en el presente trabajo fueron suministrados por el Departamento de Transporte de Nevada USA.

Finalmente a partir de la mencionada hipótesis y teniendo en cuenta que a partir de la revisión de la literatura se determinó que este es el primer acercamiento con el que se intenta calibrar modelos de simulación de flujo de tráfico vehicular mediante un algoritmo memético, se plantea la siguiente pregunta de investigación:

¿Es posible calibrar modelos de micro-simulación de flujo de tráfico vehicular CORSIM a través de un algoritmo memético?



- Los puntos que conforman la superficie ilustrada representan el espacio de posibles soluciones.
- Los puntos ○ representan maximos locales o soluciones locales, estos suponen una posible solución pero no la mejor.
- El punto ● representa el máximo global o solución global, este supone la mejor solución del conjunto de posibles soluciones de la superficie.

Nota: Dependiendo del problema sobre el cual se aplique la técnica de optimización se necesitara de un máximo o un mínimo global (representado por los puntos de color azul oscuro de la superficie).

Figura 5. Gráfica de superficie para explicar el proceso de Optimización

1.2 JUSTIFICACIÓN

El esfuerzo que se realiza por parte de varios Investigadores para tratar de mejorar los procesos de calibración de modelos de simulación es justificado principalmente por la gran herramienta que constituye el tener un modelo calibrado [16] con el que se pueda experimentar y prever con mayor exactitud comportamientos a futuro sin necesidad de esperar grandes periodos de tiempo o invertir grandes cantidades de dinero en pruebas de ensayo y error [17].

Los esfuerzos que se han realizado por el mejoramiento de los sistemas de calibración en los Modelos de Simulación de Tráfico Vehicular y en general han sido dirigidos a, mejorar tiempos, tener en cuenta múltiples parámetros y medidas simultáneamente, evadir soluciones erróneas, evitar pre-procesamientos sobre los parámetros del modelo (lo que implica más trabajo como el de realizar una pre-calibración o un estudio de sensibilidad sobre parámetros) entre otros. A continuación se exponen algunas

aproximaciones orientadas al tema de la Calibración de Modelos de Simulación de Tráfico Vehicular estos trabajos principalmente se relacionan con la rama de la Inteligencia Artificial (Algoritmos de Optimización), Matemáticas(Algoritmos Deterministas) e Investigación de Operaciones (Algoritmos Estocásticos).

Trabajos previos relacionados a la temática exponen diversas formas en las que una calibración de modelos de micro-simulación de flujo de tráfico vehicular puede ser llevada a cabo. Los algoritmos genéticos [16][18] han sido usados para calibrar diversos conjuntos de parámetros de los modelos. Sin embargo, existen meta heurísticas que han demostrado encontrar soluciones más precisas [19]. Aproximaciones determinísticas tales como el algoritmo secuencial Simplex [20] se han usado para calibrar regiones congestionadas, aunque dada la complejidad que implica realizar una calibración en la que se llegan a procesar grandes conjuntos de parámetros dependiendo del tamaño del modelo, el tiempo computacional demandado por este algoritmo puede llegar a ser muy alto [21]. Además, puede concluir en soluciones erróneas. Otras metodologías como aproximaciones estocásticas [22][23] han sido usados para calibrar características de modelos de flujo de tráfico, pero dadas sus propiedades, estos algoritmos requieren de cambios en su formulación cada vez que se aplican a un problema en particular además el establecimiento de sus parámetros puede llegar a ser un trabajo complejo [9]. Los anteriores métodos de calibración han sido usados exitosamente sobre modelos de simulación de flujo de tráfico vehicular, sin embargo no se ha encontrado una aproximación que considere todos los parámetros simultáneamente, que tome en cuenta diversas medidas, que permita calibrar de manera precisa un modelo cuyos parámetros disten de los del modelo real y que requiera de una baja intervención por parte del usuario.

Durante la revisión de la literatura se encontró que en su mayoría los trabajos de calibración fueron llevados a cabo mediante algoritmos genéticos, pues estos algoritmos aplicados a este campo han ofrecido buenos resultados [8][18].

La escogencia del algoritmo memético como herramienta de calibración se debe a que es un algoritmo con el que se han encontrado resultados favorables al ser aplicado a diversos problemas de optimización, además se encontró que ha sido aplicado sobre problemas con características similares a las del problema que presenta realizar una calibración. En diversas comparativas el algoritmo memético ha sobresalido frente a otros algoritmos heurísticos [14][19].

Guiados por la literatura en la que se expone que los algoritmos de Recocido Simulado, Búsqueda Tabú y Genéticos son algunos de los mejores algoritmos heurísticos [24][25],

se experimentará mediante instancias que combinen el algoritmo genético (que ha demostrado obtener buenos resultados como herramienta de calibración) como algoritmo de exploración y los de Búsqueda Tabú y Recocido Simulado como algoritmos de explotación. Dado que existe una amplia documentación referente a los algoritmos de las instancias anteriormente mencionadas (genético, búsqueda tabú y recocido simulado), se espera que la determinación de los valores correctos sobre los parámetros de estos algoritmos no sea muy compleja, también se espera evitar pre-procesamientos del modelo como es una sensibilización de parámetros en la que se acotan los rangos de sus valores de los parámetros con el fin de hacer el espacio de búsqueda más pequeño. Para evitar dicho pre-procesamiento sobre el modelo se empleara un algoritmo evolutivo como herramienta de exploración el cual puede encontrar soluciones a partir de un punto o conjunto de soluciones que no necesariamente sea cercano a una solución óptima.

Aunque durante la revisión de la literatura no se encontró que algoritmos meméticos hayan sido usados como herramienta de calibración sobre modelos de simulación de flujo de tráfico vehicular y aunque solo se encontraran tres artículos [26][27][28] referentes al uso de algoritmos meméticos como herramienta de calibración, se alcanza a evidenciar que la propuesta de realizar una calibración mediante un algoritmo memético es viable.

1.3 OBJETIVOS

1.3.1 Objetivo General

Proponer un algoritmo memético para calibrar modelos de micro-simulación de flujo tráfico vehicular CORSIM.

1.3.2 Objetivos Específicos

- Definir una instancia de un algoritmo memético que permita calibrar modelos de micro-simulación de flujo de tráfico vehicular CORSIM, con las siguientes características:
    - Un algoritmo de exploración: algoritmo genético, configurado con los esquemas de selección, cruce, mutación y reemplazo más usados.
    - Un algoritmo de explotación seleccionado al comparar los resultados obtenidos entre el algoritmo de recocido simulado y el de búsqueda tabú.
    - Aplicación del algoritmo de explotación seleccionado al mejor individuo de cada generación y un sistema de reinicio de población.
- Diseñar e implementar un prototipo de calibración de modelos de micro-simulación de flujo de tráfico vehicular CORSIM, basado en la instancia del algoritmo memético previamente definida.

- Evaluar la calibración de modelos de micro-simulación de flujo de tráfico vehicular realizada por el algoritmo propuesto, utilizando las directrices FHWA, y finalmente comparando los resultados obtenidos con el algoritmo SPSA.

1.4 RESULTADOS OBTENIDOS

- Monografía, actual documento en el que se describe principalmente el porqué de la investigación realizada, los objetivos, base teórica, instancia del algoritmo propuesto, experimentación, resultados y conclusiones, comparativa con SPSA, entre otros.

- Artículo, "**CALIBRATION OF TRAFFIC FLOW MODELS USING A MEMETIC ALGORITHM**".

- Prototipo Software, código fuente y ejecutable del algoritmo propuesto.

CAPITULO 2

2. ESTADO DEL ARTE

2.1 CALIBRACIÓN DE MODELOS DE SIMULACIÓN

En esta sección se muestran artículos referentes a calibración sobre diversos problemas con el fin de construir una visión general de los alcances de la calibración actualmente, los artículos aquí expuestos no están ligados a la temática puntual de esta investigación sin embargo se pueden utilizar como medio para obtener más claridad sobre la generalidad a la que pertenece el presente trabajo.

En [29] se expone el modelo cinemático de un robot. El artículo expone como la calibración del modelo es necesaria dado que existe cierto error en la vinculación de los valores de los parámetros del controlador (o nominales) y los parámetros cinemáticos con los que se mueve el robot, este error es generado principalmente por algunos de los procesos de fabricación de las partes del robot y ensamblaje de las mismas, la fórmula usada para esta calibración es llamada “product of exponentials” (POE).

En [30] se lleva a cabo la calibración de un modelo k-ε de Dinámica de Fluidos para ambientes exteriores con el fin de mejorar el trabajo de modelado de los planeadores urbanos y diseñadores de calles, la calibración es llevada a cabo mediante un método bayesiano.

En [31] se estudia la calibración de modelos de energía de construcciones E+, estos modelos contienen en sus parámetros la información referente al flujo de energía en una construcción determinada, por lo general edificios, hogares o fábricas, el artículo indica que la correcta calibración de estos modelos y el estudio de los mismos puede ayudar en la obtención de información importante para poder realizar construcciones que tengan un manejo de energía más eficiente, la calibración es llevada a cabo mediante una metodología propuesta con anterioridad por otros autores llamada Autotune.

En [32] se expone la calibración de un modelo hidrodinámico en 3D realizado en el software Delf3D de la Bahía de San Quintin en México, las medidas tenidas en cuenta durante esta calibración fueron las de batimetría, viento, elevación de la superficie del agua y velocidades del flujo de agua, la calibración es llevada a cabo mediante un

software llamado OpenDA el cual automatiza el proceso, el modelo fue calibrado con el fin de mejorar predicciones climáticas.

En [33] se exponen diferentes modelos de gravedad terrestre sobre los cuales se realiza un estudio y se concluye que el error sobre la medida de la gravedad sobre estos modelos no está bien estimada, el estudio propone una metodología para ser aplicada específicamente sobre estos modelos, la calibración se lleva cabo mediante un modelo de ajuste llamado “nonlinear condition adjustment model”.

Durante [34] se realiza un modelo matemático en el que se incorporan diversas características de un tipo de paredes usadas en diferentes construcciones pequeñas y medianas en Perú, el modelo es calibrado mediante un algoritmo genético, el estudio concluye que los resultados de los experimentos obtenidos con el modelo calibrado y los realizados sobre los muros ya construidos son cercanos.

En [35] se expone un modelo de desnitrificación-descomposición de una región de China en la que se cultiva intensivamente y en la que debido al uso masivo de fertilizantes se produce un fenómeno biológico que termina por contaminar las aguas subterráneas del lugar, el estudio finaliza al sugerir prácticas de riego para reducir efectivamente en la zona la contaminación producida por el uso masivo de fertilizantes. La calibración se basa en el proceso de un estudio de sensibilización de parámetros.

De estos artículos se pudo obtener información importante referente a la generalidad del uso de un algoritmo de optimización como herramienta de calibración, a través del estudio de las diferentes funciones de evaluación, condiciones de parada, manejo de restricciones, presentación y evaluación de resultados, se pudo extraer información importante para la realización del presente trabajo.

2.2 CALIBRACIÓN DE MODELOS DE SIMULACIÓN DE TRÁFICO VEHICULAR

Esta sección del estado del arte se expone diversos trabajos en los que se realizó calibraciones con diversos tipos de algoritmos sobre diversos tipos de modelos de simulación de tráfico vehicular, estos artículos se exponen y estudian con el fin de observar diferentes características que se compararán con el actual trabajo.

En [36] se plantea una calibración basada en un análisis de sensibilidad. Principalmente el algoritmo agrupa los parámetros similares y acota sus rangos de forma que los valores de los parámetros que alejan al modelo de simulación de los valores de campo son descartados, este proceso se realiza iterativamente hasta que los rangos de los valores

de los parámetros son aceptables para considerar el modelo calibrado, el modelo de microsimulación y microsimulador con el que trabajan es llamado MITSIMLab. El algoritmo aquí mencionado es capaz de llevar a cabo una calibración, sin embargo el proceso necesita realizar un análisis de sensibilidad acotando los rangos de los parámetros, al realizar dicho proceso se pueden descartar valores que pueden brindar más precisión sobre la calibración, además cuando este tipo de procesos se desea realizar sobre un modelo que tiene grandes cantidades de parámetros el proceso se hace inviable. El algoritmo memético que se propone como herramienta de calibración realiza una búsqueda intensiva sobre diversos grupos de parámetros con sus rangos abiertos lo que permite encontrar soluciones más precisas pues no se descartan las posibles soluciones que se descartan al cerrar los rangos de los parámetros.

En [37] se expone una metodología propia de calibración en línea de modelos de buses sobre el simulador Aimsun, la calibración en línea trata modelos que en tiempo real consiguen los valores para sus parámetros a través de diversos sensores que extraen la información directamente de campo hacia una central, en este caso el estudio se centra en el impacto de las estrategias prioritarias de señales y las emisiones de buses sobre algunas vías en Londres. Este tipo de calibración de modelos es uno de los más modernos y fiables pues utiliza información generada en tiempo real que se actualiza en una central constantemente, sin embargo los costos de generar este tipo de información en tiempo real son elevados y por lo tanto pocas ciudades y vías disponen de la infraestructura necesaria (sensores y centrales) para lograr este tipo de calibración sobre sus modelos de simulación, así, la calibración que se lleva a cabo en el presente trabajo denominada calibración “off-line” se hace necesaria para la mayoría de modelos a los que los diversos departamentos de transporte tienen acceso.

La investigación mostrada en [38] se centra en la creación de un microsimulador de tráfico vehicular de una sola vía mediante la herramienta Matlab, la calibración es realizada mediante un algoritmo iterativo poco descrito en el documento. El modelo expuesto durante este trabajo es un modelo básico que contiene pocos parámetros con los que difícilmente se podrían representar las complejas variables que requiere un modelo que trata de acercarse a reflejar el comportamiento de fenómenos de la vida real.

En [39] se propone un simulador de tráfico vehicular construido a partir de autómatas celulares, la calibración se realiza sobre modelos en línea y el algoritmo usado para realizar la calibración es un algoritmo genético. El artículo aquí mencionado expone que los autómatas celulares como individuos sobre un microsimulador llegan a representar de manera adecuada el comportamiento de los vehículos y como ya se había mencionado en un anterior articulo la calibración en línea funciona de manera precisa, sin embargo como

se mencionó anteriormente, los costos que implica el tipo de calibración usado son muy altos limitando la aplicación de este tipo de calibración a pocos sitios.

En [40] se realiza una calibración a partir de la información conjunta de dos simuladores, uno de tráfico vehicular llamado KTH-TPMA y otro simulador de emisiones, la calibración se realiza haciendo uso del algoritmo SPSA. El algoritmo SPSA puede calibrar modelos, pero el proceso de establecer los valores de parámetros necesarios para el correcto funcionamiento de este algoritmo puede llegar a ser complejo, además la solución de la que parte el algoritmo debe ser una solución cercana haciendo que el modelo deba pasar por una sensibilización de parámetros. Durante el presente trabajo se realizará una comparativa con un trabajo en el que se utilizó el SPSA como algoritmo de calibración sobre modelos CORSIM.

En [41] se expone la calibración de un modelo de microsimulación de tráfico vehicular, el modelo es construido en un popular software de microsimulación de flujo de tráfico vehicular llamado VISSIM, la calibración es llevada a cabo exitosamente mediante un algoritmo de redes neuronales. Sin embargo el algoritmo propuesto solo puede tomar en cuenta una medida a la vez lo cual hace que ciertos estudios sean inviables, por ejemplo durante el presente trabajo se pueden determinar embotellamientos sobre los modelos calibrados al tener en cuenta simultáneamente dos de las medidas más importantes para poder determinar cuándo se presentan embotellamientos en un modelo; las medidas son, velocidad y volumen del flujo vehicular. Los números de entrada de estos algoritmos son fijos, el algoritmo memético propuesto es capaz de usar un número de entrada variable pues los diferentes modelos presentan un número diferente de entradas (parámetros).

2.3 CALIBRACIÓN MEDIANTE ALGORITMOS MEMÉTICOS

En esta sección se exponen los artículos que se encontraron durante la revisión de la literatura referentes a calibración mediante algoritmos meméticos en general, se realiza con el ánimo de mostrar que la calibración de modelos de simulación de flujo de tráfico vehicular puede ser viable.

En [26] se realiza un proceso de calibración mediante un algoritmo memético para un transductor de fuerza de rueda que es básicamente un sensor multidimensional de fuerzas el cual sirve para obtener información relevante en la investigación de la vibración, suspensión, dinámica de ruedas, transmisión y sistema de frenado de un vehículo. El artículo expone que al poner el sensor sobre la rueda de un automóvil, los valores iniciales cambian debido a que desde ese momento diferentes fuerzas empiezan a ejercerse en él. La solución presentada básicamente hace que el algoritmo memético

realice una búsqueda de valores para que los datos de entrada y salida del sensor sean coherentes de tal forma que este brinde valores con mayor precisión en sus mediciones. A pesar de que los algoritmos, función de evaluación y otros componentes usados en el presente trabajo para instanciar el algoritmo memético son diferentes a los usados en el artículo, el esquema general del algoritmo y varias características fueron estudiadas con el fin de ser adaptadas al actual trabajo.

En [27] se propone un algoritmo memético llamado "shuffled frog-leaping (SFLA)" el cual es desarrollado con el fin de resolver problemas de optimización combinatoria, los autores además demuestran cómo el algoritmo puede ser usado para otros fines entre los cuales está el de realizar calibraciones. A pesar de que el artículo no se centra en solucionar un problema de calibración de modelos, se expone la calibración de un modelo poco complejo de aguas subterráneas con el fin de demostrar que el algoritmo “shuffled frog-leaping” es capaz de solucionar diversos problemas de optimización. Este artículo fue el único que se encontró durante la revisión de la literatura en el que se realizó la calibración de un modelo de simulación. Aunque el algoritmo aquí propuesto funciona de manera adecuada al ser usado como herramienta de calibración tiene el problema de requerir de la modificación de sus parámetros cada vez que se necesite calibrar un modelo y la determinación de sus parámetros es compleja, durante el presente trabajo se realizó un esfuerzo para que el conjunto de parámetros requeridos por las instancias del algoritmo memético fuera estático y se usaron algoritmos fuertemente documentados con el fin de que su construcción y parametrización fuera lo más sencilla posible.

En [28] el rendimiento y emisión de gases de un vehículo son calibrados mediante un algoritmo memético, este busca valores óptimos para los parámetros como el flujo de masas de aire, velocidad del motor, entre otros, de tal manera que el vehículo tenga un rendimiento y emisión de gases deseado. El artículo presenta diversas instancias de un algoritmo memético en el que se usan algoritmos genéticos como herramienta de calibración, entre las conclusiones del trabajo se muestra que los algoritmos genéticos son una buena herramienta de exploración para el problema que presenta la calibración, conclusión que ayudo en el proceso de la escogencia de la instancia del algoritmo memético que se plantea para calibrar modelos de simulación de tráfico vehicular CORSIM.

A partir de los artículos expuestos se puede concluir que los algoritmos meméticos pueden ser buenos para realizar la tarea de calibración que se requiere, sin embargo se muestra como la calibración de diferentes objetos o modelos presentan diferentes necesidades que para ser suplidas requieren de un minucioso estudio.

CAPITULO 3

3. MARCO TEORICO

3.1 INTRODUCCIÓN

Este marco teórico se construye con el fin de exponer diversos conceptos que podrían llegar a ser necesarios para mejorar el entendimiento del presente documento, para exponer cada uno de estos conceptos se realizará una descripción general que se acompañara con artículos a manera de referencia, dichos artículos pueden ser utilizados para encontrar información más detallada del concepto expuesto.

3.2 MICROSIMULACIÓN

En el año 1957 el economista Guy Orcutt propuso la microsimulación como una forma de simulación basada en computador, esta forma de simulación expuso como principal característica el modelado de procesos a nivel individual o micro y la obtención de conclusiones a nivel macro al formar una generalidad de los comportamientos a través de la suma de los resultados obtenidos al procesar el comportamiento de cada uno de los individuos del modelo. Durante los años 1950 a 1990 este tipo de modelos no fue popularmente usado pues el esfuerzo de hacerlos y mantenerlos era alto, no fue sino a partir del año 1990 que este tipo de modelos empezó a hacerse popular al haber demostrado buenos resultados al ser aplicado en el ámbito económico en los procesos de cálculo de impuestos y análisis de transferencias, además a partir de 1990 el poder computacional y la facilidad de obtención de información mejoraron notablemente haciendo que este tipo de modelos fuera cada vez más aceptado [5]. La Figura 6 muestra la representación gráfica de un modelo de microsimulación en 3d en el que se observan diversos tipos de vehículos (individuos) calles y demás componentes de una vía.

Figura 6. Gráfica de modelo de microsimulación 3d

3.3 MODELOS DE MICROSIMULACIÓN DE TRÁFICO VEHICULAR CORSIM Y SIMULADOR CORSIM

CORSIM (CORridor SIMulation) [3], software propietario, fue creado en la Universidad de Florida en el año 1998, actualmente se encuentra en la versión 6.3, su funcionamiento es de tipo caja negra en el que se tiene información referente al formato de los archivos de entrada (extensión .trf) y salida (extensión .out) con los que trabaja, pero al ser un software propietario se desconocen los procesos que se ejecutan al correr una simulación y por lo tanto se desconoce cómo se forman los datos de salida a partir de los datos de entrada.

Los archivos de entrada tienen extensión ".trf", (en las figuras 1 y 2 se exponen los elementos que los constituyen) estos archivos contienen toda la información del modelo y está compuesta por diversos grupos de parámetros llamados "record type" los cuales contienen la información necesaria para representar diversos comportamientos a través del modelo y su simulación, un ejemplo sencillo expuesto anteriormente es el del record type 21 el cual contiene la información del porcentaje de vehículos que seguirán derecho, cruzarán a la izquierda, cruzarán a la derecha o tomarán una diagonal (Este record type se expone en la figura 2) durante una simulación sobre dos determinados links.

Los archivos de salida contienen diversos grupos de medidas formadas a través de la suma de los comportamientos de los individuos (vehículos) en el modelo, dichas medidas son obtenidas de las diferentes intersecciones del modelo, estas intersecciones son llamadas nodos los cuales a su vez se forman a partir de la unión de dos vías (links). La información contenida en estos archivos se encuentra sobre diferentes tablas que muestran a manera de resumen las estadísticas obtenidas de la simulación sobre diferentes nodos, por ejemplo se puede observar la velocidad promedio que hubo durante la simulación entre dos links del modelo.

CORSIM es un microsimulador compuesto por la aplicación NETSIM (NETwork SIMulation) con la que se simulan escenarios de tráfico urbano y FRESIM (FREway SIMulation) que se encarga de simular el tráfico en autopistas, cuando un modelo es simulado las aplicaciones NETSIM y FRESIM trabajan conjuntamente [42]. El microsimulador CORSIM es el núcleo de un framework llamado TSIS el cual permite diseñar los modelos a través de su módulo TRAFED, Figura 7, y observar el comportamiento de una simulación a través del módulo TRAFVU, Figura 8.

3.4 CALIBRACIÓN DE MODELOS DE SIMULACIÓN

La calibración de modelos permite reducir la incertidumbre de los parámetros y, por lo tanto, la incertidumbre en los resultados de una simulación. Durante la calibración de un modelo, parámetros seleccionados se varían dentro de límites razonables hasta que se obtiene una correspondencia suficiente entre una o varias variables de salida del modelo y las mediciones (de campo) respectivas [43].

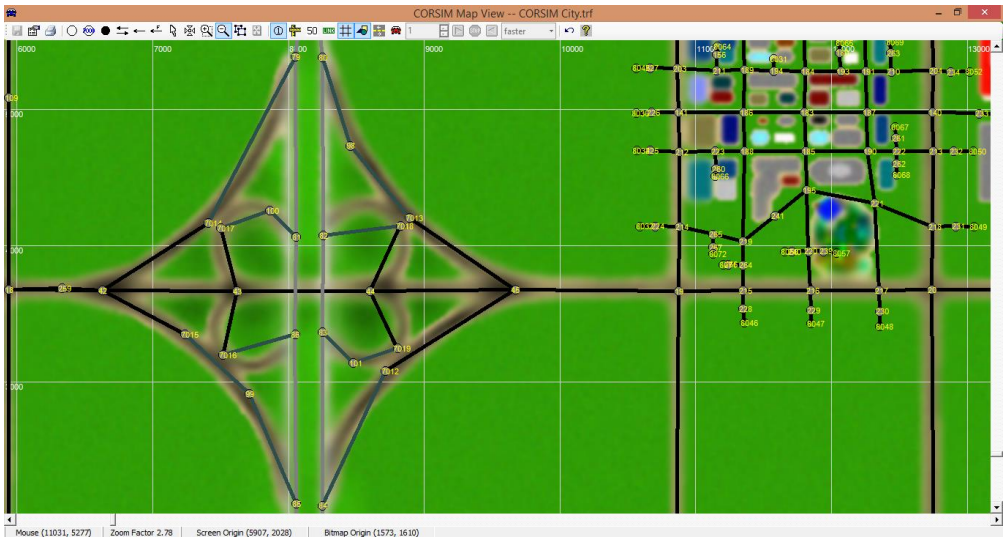


Figura 7. Modelado mediante TRAFED

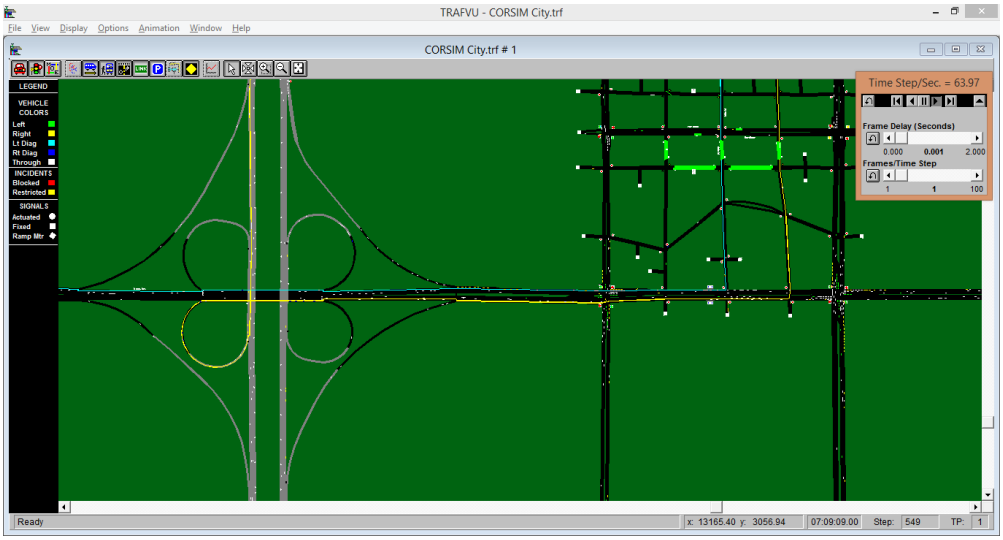


Figura 8. Simulación mediante TRAFVU

Existen diversas mediciones que se pueden obtener al simular un modelo, por ejemplo sobre un modelo de recursos hídricos se puede obtener el flujo total de agua de lluvias que recibió un rio en un determinado mes del año, información que llega a ser importante

para investigar el comportamiento del caudal a través de los años. Para poder hacer que dicha medida obtenida de una simulación sea fiable se requiere de un modelo calibrado.

El proceso de calibración se hace necesario dada la incertidumbre de los valores iniciales que deben ser fijados sobre los parámetros de un modelo al ser construido. Generalmente dicha extracción de medidas tiene una alta complejidad, por ejemplo sobre los modelos de simulación de tráfico vehicular el tiempo de reacción de frenado del conductor de un vehículo es difícil de ser medido, la dificultad crece cuando sobre un modelo hay un gran número de individuos y para cada uno de estos individuos se requiere obtener múltiples medidas que tienen una alta dificultad de ser extraídas en campo, además dichas medidas deberían ser extraídas simultáneamente para todos los individuos pues los modelos habitualmente se procesan por periodos de tiempo.

Generalmente los modelos de simulación que presentan una alta complejidad al ser construidos, tienen como fin representar de la forma más precisa posible los comportamientos que se observan en la zona en la que se construyó el modelo por lo tanto la incertidumbre sobre los valores de los parámetros debe minimizarse a través del proceso de calibración.

La comparación que se lleva a cabo en una calibración entre los datos de salida del modelo simulado y los datos extraídos de campo, no es diferente a la comparación entre las medidas obtenidas en una simulación y los datos medidos en campo, tal comparativa decide si la calibración se ha realizado exitosamente pues en el proceso de calibración debe existir una medida que establezca la diferencia entre dichos conjuntos de datos, además se debe conocer hasta qué punto se desea minimizar la diferencia entre estos conjuntos lo cual podría servir para aminorar el tiempo que el algoritmo requiere para llevar a cabo la calibración.

Al reducir la incertidumbre en los resultados de una simulación el proceso de calibración permite que los datos obtenidos al realizar diversas experimentaciones sobre el modelo calibrado retornen resultados más fiables. Existe un sin número de aplicaciones para un modelo calibrado, por ejemplo al realizar la calibración de un modelo climático (temperatura, humedad, precipitaciones) la predicción del comportamiento del clima puede ser más acertada.

Un sin número de algoritmos de optimización han sido usados en la calibración de modelos de simulación, estos algoritmos se encargan principalmente de realizar el proceso de variación de los parámetros y de validar si la correspondencia entre las variables de salida del modelo y los datos de campo es suficiente.

3.5 CALIBRACIÓN DE MODELOS DE MICROSIMULACIÓN DE TRÁFICO VEHICULAR CORSIM

La calibración de estos modelos presenta en general las mismas características y dificultades descritas en el párrafo anterior [44][8], entre algunas de las principales características y/o dificultades que se presentan al realizar una calibración sobre estos modelos se encuentra el de tener en cuenta diversas restricciones en los parámetros, en las que algunos conjuntos de “record type” tienen que guardar coherencia con la información que requieren, por ejemplo sobre el record type 21 mencionado anteriormente se debe validar que la suma de sus partes de como resultado 100 pues es un porcentaje, es decir la suma de porcentajes de vehículos que giran hacia la derecha, izquierda, siguen derecho o toman una diagonal debe ser 100.

Otra dificultad que se presenta al calibrar estos modelos es que dependiendo de su tamaño el tiempo de ejecución de la simulación necesaria para poder obtener los datos de salida y compararlos con los datos de campo puede ser muy alto y por ende el tiempo de calibración sea alto también, por esto el algoritmo de calibración debe ser optimizado con el fin de lograr una calibración con el menor número de corridas del simulador posible.

En el presente trabajo se plantea calibrar estos modelos teniendo en cuenta todos los parámetros calibrables del modelo simultáneamente y tomando en cuenta las medidas de velocidad y volumen que genera como resultado la ejecución de un modelo CORSIM (existen otras medidas como por ejemplo de emisiones de gases sin embargo no se encontró este tipo de información en los datos de campo que fueron suministrados por la Federeal Highway Administration FHWA de Estados Unidos), acerca de estos datos se conoce que los datos fueron obtenidos a partir de información extraída de sensores especializados y observación en campo, el índice que se utilizó para medir la diferencia entra los datos de salida del simulador CORSIM y los datos de campo, fue la media cuadrada normalizada.

En los trabajos encontrados durante la revisión de la literatura se pudo observar que un algoritmo frecuentemente usado para realizar diferentes trabajos de calibración sobre este tipo de modelos es el algoritmo genético [17][44][18][45], La Figura 9 describe el proceso de calibración de un modelo CORSIM.

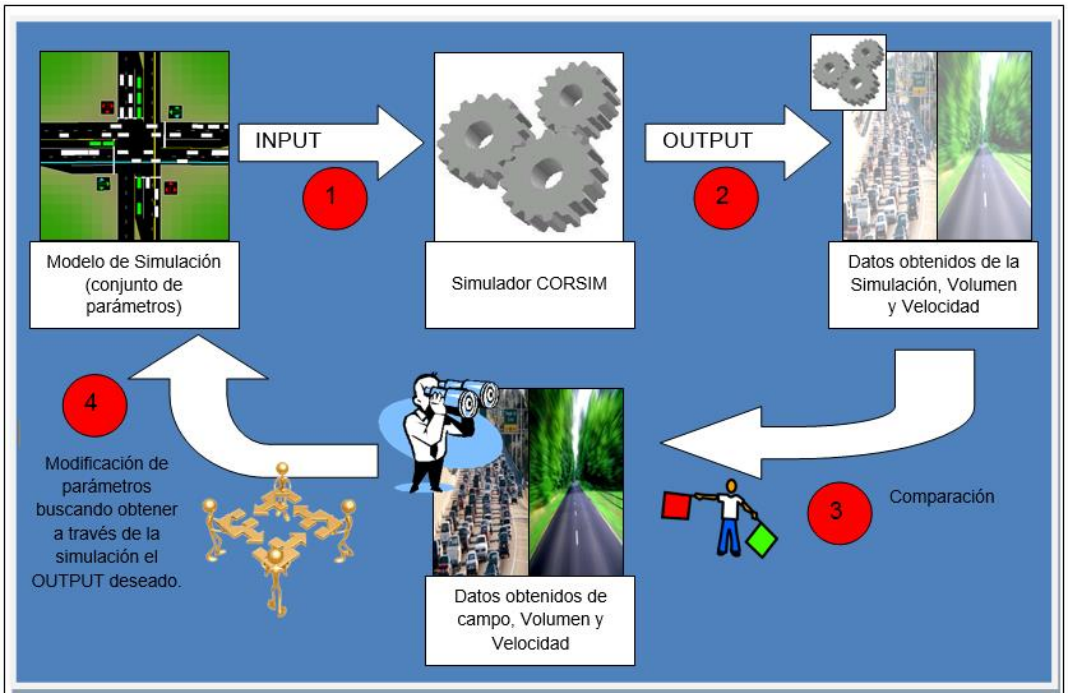


Figura 9. Gráfica que explica el proceso de calibración

3.6 ALGORITMO DE OPTIMIZACIÓN

En general, un algoritmo de optimización es un método numérico o algoritmo capaz de encontrar un valor (x) tal que f(x) es tan pequeño (o tan grande) como sea posible, para una función f dada, posiblemente con algunas restricciones en x, donde x puede ser un vector escalar o vector de valores continuos o discretos [46]. El valor de x que se desea encontrar se halla sobre un determinado espacio de búsqueda que está determinado por el rango de valores que las variables del vector o escalar x pueden tomar, los algoritmos de optimización requieren de al menos una función que indique en qué medida el valor de x que se evalúa es cercana al valor que se requiere, dicha función es llamada función de evaluación, en este caso f(x). Muchos de los algoritmos de optimización realizan iteraciones en las que a través de diversos métodos varían y evalúan diferentes conjuntos de valores del vector x hasta que la función de evaluación determina que los valores del vector x son lo suficientemente cercanos a los requeridos, uno de los problemas más frecuentes que tienen estos algoritmos es el de encontrar la forma de variar los valores en los vectores de forma que dichos valores se acerquen cada vez más a la solución deseada ya que de no ser así los algoritmos requerirían de tiempos de ejecución demasiado largos pues el espacio de búsqueda en problemas de optimización comúnmente es grande, así los algoritmos de optimización enfrentan un nuevo problema al tratar de encontrar este conjunto de datos a través de mejoras pues los algoritmos pueden encontrar puntos que consideran como la mejor solución sin embargo el espacio

de búsqueda puede contener aún mejores soluciones y el algoritmo puede no encontrarlos, esto se debe a que frecuentemente los algoritmos de optimización funcionan a través del concepto de vecinos o vecindades que pueden determinarse como zonas o individuos “cercanos” a una buena solución sin embargo el algoritmo puede quedar atrapado al no encontrar una mejor solución alrededor de estas zonas o individuos. La figura 5 expone como funciona un algoritmo de optimización y ayuda a entender el problema de los máximos y mínimos locales, dado que la definición y funcionamiento de un algoritmo de optimización es complejo, se aconseja observar la referencia [47] en la que se expone una gran cantidad de algoritmos de optimización y diversos conceptos de los mismos.

3.7 ALGORITMO DE OPTIMIZACIÓN COMO HERRAMIENTA DE CALIBRACIÓN DE MODELOS DE SIMULACIÓN

Partiendo de la definición de Algoritmo de Optimización y de Calibración de Modelos de Simulación, x se define el conjunto de parámetros tal que f(x) como la medida de error o “diferencia” entre las variables de salida del modelo y las medidas de campo sea lo más pequeño posible.

3.8 ALGORITMO MEMÉTICO

Los Algoritmos Meméticos son una familia de mataheurısticas que intentan aunar ideas y conceptos de diferentes técnicas de resolución, como por ejemplo Algoritmos Evolutivos y Búsqueda Tabú [14]. Estos algoritmos fueron propuestos por Moscato en el año de 1989, el adjetivo “memético” viene del inglés “meme” usado por R. Dawkins para designar al análogo del gen en el contexto de la evolución cultural, sin embargo el propósito de esta terminología tiene como fin hacer explícito que se difumina la inspiración puramente biológica y se opta por modelos más genéricos en los que se manipula se aprende y se transmite información, en general la definición de estos algoritmos es abierta dado que se forma a partir del conocimiento de que cada problema de optimización presenta diferentes características. Los algoritmos meméticos son metaheuristicas basadas en población, principalmente estos algoritmos funcionan a partir de la colaboración de dos componentes, uno de estos es el de exploración en el cual un algoritmo por lo general evolutivo realiza una búsqueda sobre todo el espacio de búsqueda en el que intenta hallar zonas que puedan contener una buena solución, luego, la otra parte el algoritmo de explotación realiza una búsqueda intensiva tratando de encontrar el mejor resultado que se pueda obtener sobre la zona escogida por el algoritmo de exploración. Finalmente un algoritmo memético en su definición más sencilla puede verse como una colección de agentes que realizan una exploración autónoma del espacio de búsqueda [48].

3.9 ALGORITMO GENÉTICO

Los algoritmos genéticos fueron propuestos por primera vez en el año de 1970 por John Henry Holland, fueron llamados de esta forma dado que su funcionamiento está inspirado en la evolución biológica y su base genética. Su funcionamiento parte de un conjunto de posibles individuos o soluciones llamado fenotipo (o población), los individuos son generalmente codificados como vectores llamados cromosomas, cada una de las posiciones de estos vectores es llamada gen, su metodología se basa en ir de una población de "cromosomas" a una nueva población mediante el uso de un tipo de "selección natural", a través de operadores de cruce inspirados en genética como son los de mutación y cruce, cada nueva población obtenida a partir de las iteraciones del algoritmo estará compuesta por nuevos individuos que generalmente representan mejores soluciones en un problema de optimización [49]. Existen diversas formas de realizar los procesos de selección, cruce y mutación, generalmente al hacer uso de un algoritmo genético se debe realizar un estudio para determinar cuáles son los operadores indicados para que el algoritmo funcione sobre un determinado problema de optimización de forma adecuada.

Al igual que otros algoritmos de optimización, este algoritmo requiere de una función de evaluación que le permita determinar qué tan buena es la solución representada por cada individuo de la población con la que trabaja el algoritmo, también se hace necesario definir una función de parada o un numero de iteraciones máximo del algoritmo con el fin de optimizar su proceso y evitar ciclos infinitos de búsqueda.

3.10 ALGORITMO DE BÚSQUEDA TABÚ

El algoritmo metaheurístico de búsqueda tabú fue propuesto por primera vez por Fred Glover en el año de 1986, este algoritmo tiene la capacidad de resolver problemas de optimización combinatoria y es considerado como un buen algoritmo de búsqueda local.

El algoritmo de búsqueda tabú es llamado de esta forma dado que tiene un mecanismo que le permite descartar soluciones ya visitadas y/o puede excluir soluciones que presenten ciertos atributos evitando que el algoritmo procese más información de la necesaria, este mecanismo es llamado la “lista tabú”, y es sobre la cual se registran soluciones ya visitadas y restricciones sobre los individuos no deseados por el algoritmo.

El funcionamiento de este algoritmo se basa en la definición de vecindades sobre las cuales puede moverse de un punto (posible solución) en una vecindad (conjunto de soluciones cercanas) a otro a través de iteraciones que le permiten aproximarse a una mejor solución existente en la misma vecindad, el procedimiento se repite hasta que un

determinado criterio de parada se ejecute en el algoritmo. Una dificultad que frecuentemente presentan los algoritmos que funcionan a partir de la definición de vecindades, es determinar el tamaño de la vecindad y que solución es considerada como una solución vecina [50][51].

3.11 ALGORITMO DE RECOCIDO SIMULADO

El enfriamiento simulado es un método probabilístico propuesto por Kirkpatrick, Gelett y Vecchi (1983) y Cerny (1985) para encontrar el mínimo global de una función. El algoritmo emula el proceso físico en el que un sólido es lentamente enfriado de modo que cuando finalmente su estructura llegue a estar "congelada" ocurra con un gasto mínimo de energía [52]. Su funcionamiento es iterativo, principalmente en cada una de las iteraciones el método de recocido simulado evalúa un pequeño número de individuos (pertenecientes a un vecindario) y a partir de la evaluación obtenida, determina probabilísticamente si es necesario cambiar de vecindario, de esta forma el algoritmo se desplaza a través de diversas zonas en el espacio de búsqueda, mecanismo con el cual realiza su búsqueda del mínimo global hasta que un criterio de parada se cumpla.

Capítulo 4

4. ENFOQUE METODOLOGICO

4.1 INTRODUCCIÓN

La metodología utilizada durante el desarrollo del presente trabajo se basa en el Patrón de Investigación Iterativa [53], dicho patrón fue escogido debido a que algunas de sus características facilitaban y se amoldaban a las necesidades que el desarrollo de la presente investigación planteaba pues al ser una primera aproximación o un trabajo de naturaleza exploratoria (en los que se desconoce con exactitud cómo desarrollar los objetivos planteados), se concluyó que era conveniente el uso de una metodología que permitiera un desarrollo iterativo e incremental de los diferentes objetivos planteados.

4.2 DESCRIPCIÓN

El patrón utilizado cuenta con 4 fases principales: Observación (del problema), identificación (del problema), desarrollo (de la solución) y prueba (de la solución), además cuenta con una fase de documentación y divulgación transversal a las fases previas.

En el desarrollo del primer objetivo específico se llevaron a cabo tres ciclos, con el fin de determinar la instancia del algoritmo memético a usar como herramienta de calibración (ciclo 1: Configuración del algoritmo de exploración, ciclo 2: Configuración de algoritmo de explotación, ciclo 3: Configuración general del algoritmo memético).

- **Observación:** A partir de la revisión de la literatura se encontró documentación que ayudo a determinar las instancias necesarias para que el algoritmo memético funcionara de manera adecuada sobre el problema de calibración planteado, además se encontró información relevante para la parametrización de sus algoritmos de exploración y explotación (Algoritmos genéticos, Recocido Simulado y Búsqueda Tabú).

- **Identificación:** A partir de algunos trabajos expuestos en el estado del arte se determinó que los algoritmos genéticos tenían un gran potencial para ser usados como herramienta de exploración en el algoritmo memético, otros documentos expuestos en la misma sección demuestran como los algoritmos de búsqueda tabú y recocido simulado han obtenido buenos resultados al ser aplicados sobre diversos problemas de optimización convirtiéndolos en buenas opciones para ser usados en

esta primera aproximación en la que se plantea hacer uso de algoritmos meméticos como herramienta de calibración.

Los ciclos ejecutados para desarrollar el primer objetivo específico determinaron la instanciación y parametrización necesaria para solucionar el problema planteado. Las instancias del algoritmo memético que se determinaron tienen como componentes:

a) Un algoritmo genético (como herramienta de exploración) combinado con un algoritmo de búsqueda tabú (como herramienta de explotación).
b) Un algoritmo genético (como herramienta de exploración) combinado con un algoritmo de recocido simulado (como herramienta de explotación).

La documentación obtenida a partir de la revisión de la literatura ayudo a determinar los valores con los que se pudo realizar diversos experimentos con el fin de afinar los parámetros de forma que permitieran al algoritmo realizar calibraciones de modelos de simulación de flujo de tráfico vehicular CORSIM.
En el desarrollo del segundo objetivo específico se llevaron a cabo 3 ciclos, en cada uno se desarrollaron y adaptaron los algoritmos genético, búsqueda tabú y recocido simulado mediante el proceso de desarrollo ágil llamado Programación Extrema.

- **Observación:**

  La documentación que presenta las historias de usuario y diagramas de clases en los que se expone el proceso de planeación de la construcción del algoritmo se encuentra en el anexo 1 del presente documento.

- **Identificación:** componentes del algoritmo de calibración.

a) Procesos que permitieran la extracción de parámetros de los modelos (datos de entrada)
b) Medios para que los parámetros del modelo pudieran ser modificados y entregados al simulador.
c) Ejecución del simulador en paralelo (para aminorar tiempos de calibración) desde el prototipo para generar datos de salida a partir del procesamiento de uno o varios modelos.
d) Medios para la extracción de datos de los archivos de salida generados por el simulador.
e) Procesos que permitieran la extracción de datos de campo suministrados.

f) Procesos que permitieran generar un índice de error mediante la comparativa entre los datos de salida de la simulación y los datos de campo con la aplicación de la media cuadrática normalizada.
g) El algoritmo memético (cada instancia) debe tener acceso a cada una de los procesos nombrados y ejecutarlos iterativamente.
h) El algoritmo memético tiene como función de parada la combinación de un máximo número de iteraciones y el cálculo de un índice llamado GEH que determina si un modelo esta calibrado.
i) El desarrollo del prototipo se realizara mediante un patrón de capaz expuesto en el anexo 1.

- **Desarrollo:** La descripción de las iteraciones de desarrollo del algoritmo se encuentra en el anexo 1 del presente documento.

- **Pruebas:** Diversos experimentos fueron llevados a cabo con el fin de determinar el correcto funcionamiento de las instancias del algoritmo memético, en dichas pruebas se cambiaron los parámetros dentro de límites razonables obtenidos de la literatura para afinar los algoritmos, la función de evaluación fue parte determinante ya que verifica matemáticamente que los resultados obtenidos sean precisos, las pruebas determinaron que el algoritmo logro realizar exitosamente el proceso de calibrar un modelo de simulación de tráfico vehicular CORSIM.

El desarrollo del tercer objetivo específico se realizó en un ciclo con las siguientes actividades.

- **Desarrollo:** Se realizó una comparativa a partir de los resultados obtenidos de 3 experimentación mediante las dos instancias del algoritmo propuesto en el presente trabajo, un algoritmo genético y el algoritmo SPSA siendo usados como herramienta de calibración, esta comparativa se realizó tomando en cuenta los tiempos de ejecución y precisión de los 4 algoritmos. En el presente capitulo se expondrá en detalle cada experimento.

- **Pruebas:** Los resultados obtenidos a partir de la ejecución de los 4 algoritmos como herramienta de calibración en 3 diferentes modelos fue organizada y se observó si los algoritmos pudieron calibrar los modelos, en cuento tiempo lo hicieron y que precisión se obtuvo con cada uno.

A partir de la información obtenida durante las fases de observación de la presente metodología se pudo apreciar que se requería de un lenguaje de programación que contara con algunas características especiales, entre ellas:

- Facilidad de ejecución de funciones en paralelo (para optimizar tiempos de ejecución de la calibración).
- Manejo de archivos (múltiples tareas del prototipo requieren de la manipulación de archivos de texto sobre los que se encuentran diversos datos de vital importancia).
- Óptimo funcionamiento y Facilidad de manipulación de diversas estructuras de datos (Miles de datos deben ser ordenados y mantenidos en memoria durante el proceso de calibración).
- Tener funciones matemáticas fáciles de aplicar sobre diversas estructuras de datos (El algoritmo de calibración tiene fuertes componentes matemáticos).
- Orientado a objetos (Con el fin de crear una arquitectura con la que se pueda llevar un orden adecuado en la construcción del software y a través del encapsulamiento lograr separar diversas funcionalidades y variables).
- Que tenga un framework avanzado (La construcción del mencionado prototipo es bastante compleja).

Dadas las razones anteriormente expuestas se optó por utilizar el lenguaje de programación Java.

4.3 DISEÑO DE LOS EXPERIMENTOS

4.3.1 Introducción

Con el objetivo de poner a prueba la solución propuesta se han definido 3 experimentos con diferentes niveles de complejidad (baja, media y alta), la complejidad de cada experimento fue definida en términos del consumo de recursos por parte de los algoritmos (SPSA, GA y las dos instancias del Alg. Memético) y su comportamiento al ser usados como herramienta de calibración (o en qué medida lograron el objetivo de calibrar los modelos), en general se observó que la complejidad fue baja en los modelos que tenían entre 0 y 99 enlaces, media entre 100 y 300 enlaces y alta en los modelos de más de 300 enlaces, todo lo anterior, en relación al equipo con el que se trabajó durante la presente investigación mencionado en la sección 4.4.1.

La valoración de los resultados obtenidos durante la experimentación se hace mediante diversas medidas matemáticas (NRMS y GEH), dichas medidas y resultados se exponen a profundidad posteriormente en el presente documento.

Como se mencionó previamente los datos de campo y los modelos fueron provistos por NDOT, Nevada Department of Transportation (Deparamento de transportes de nevada). El conjunto de parámetros a calibrar son los relacionados con comportamiento del conductor y rendimiento, que en [3] se definen como los parámetros calibrarles de un modelo. Es importante aclarar que para dicha calibración se toman solamente las salidas de simulación relacionadas con volumen de tráfico vehicular vph (vehículos por hora) y velocidad de los vehículos mph (millas por hora).

Los modelos, selección de parámetros y datos de campo referentes a los experimentos aquí descritos pueden ser encontrados en los anexos de este documento, esta información sumada a la expuesta en el presente documento asegura la replicabilidad de los experimentos aquí expuestos.

4.3.2 Experimento 1: Modelo complejidad baja: Mac Trans network

Este experimento facilito la afinación de la mayoría de los operadores de las instancias propuestas pues su consumo de recursos fue bajo, lo que permitió realizar varias pruebas en un tiempo acorde al tiempo planteado en el cronograma del presente trabajo.

La finalidad que tuvo la ejecución de este experimento fue la de recolectar datos que evidenciaran el comportamiento de las instancias propuestas (además del alg. GA y SPSA) al calibrar un modelo de baja complejidad.

La Figura 10 muestra la descripción de modelo a calibrar. Número de enlaces: 20.

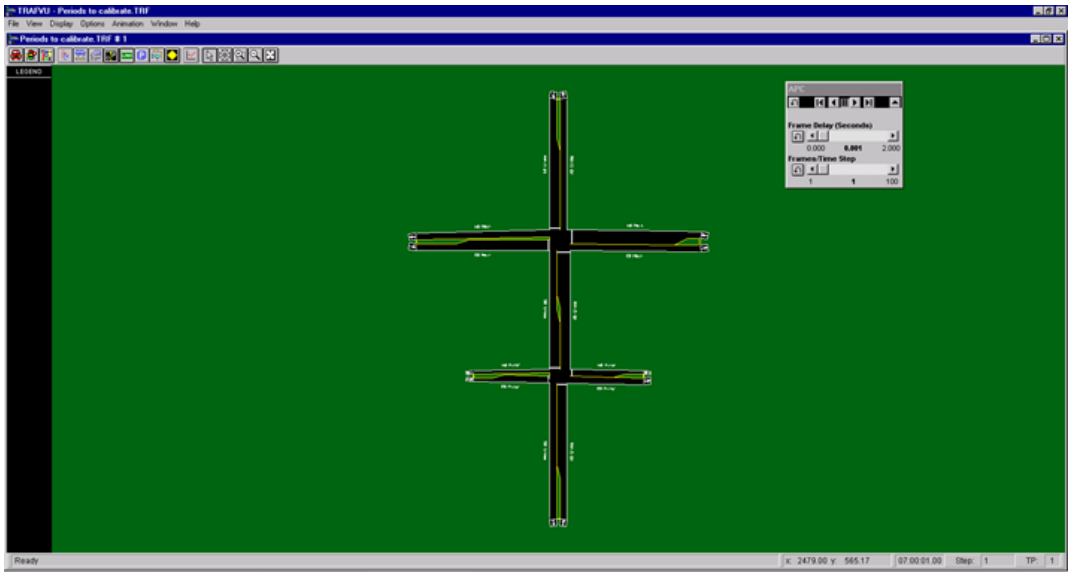
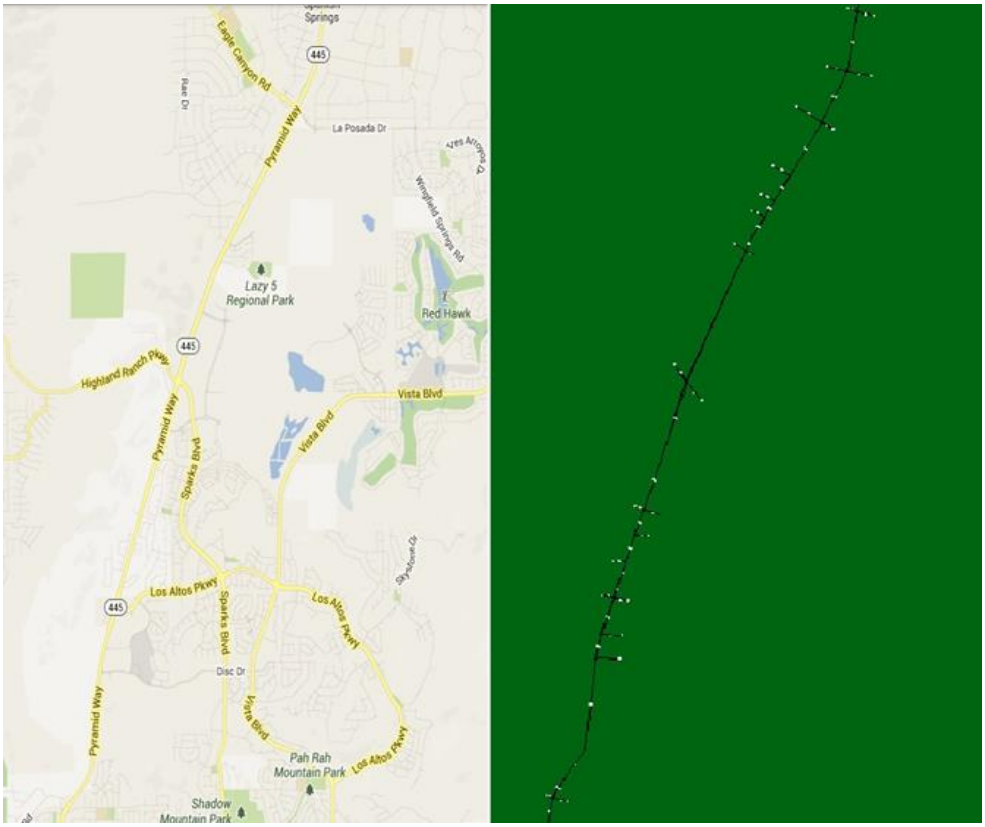


Figura 10. Modelo ejemplo del software TSIS, nombre: Mac Trans.

4.3.3 Experimento 2: Modelo complejidad media: Reno network



(a) (b)

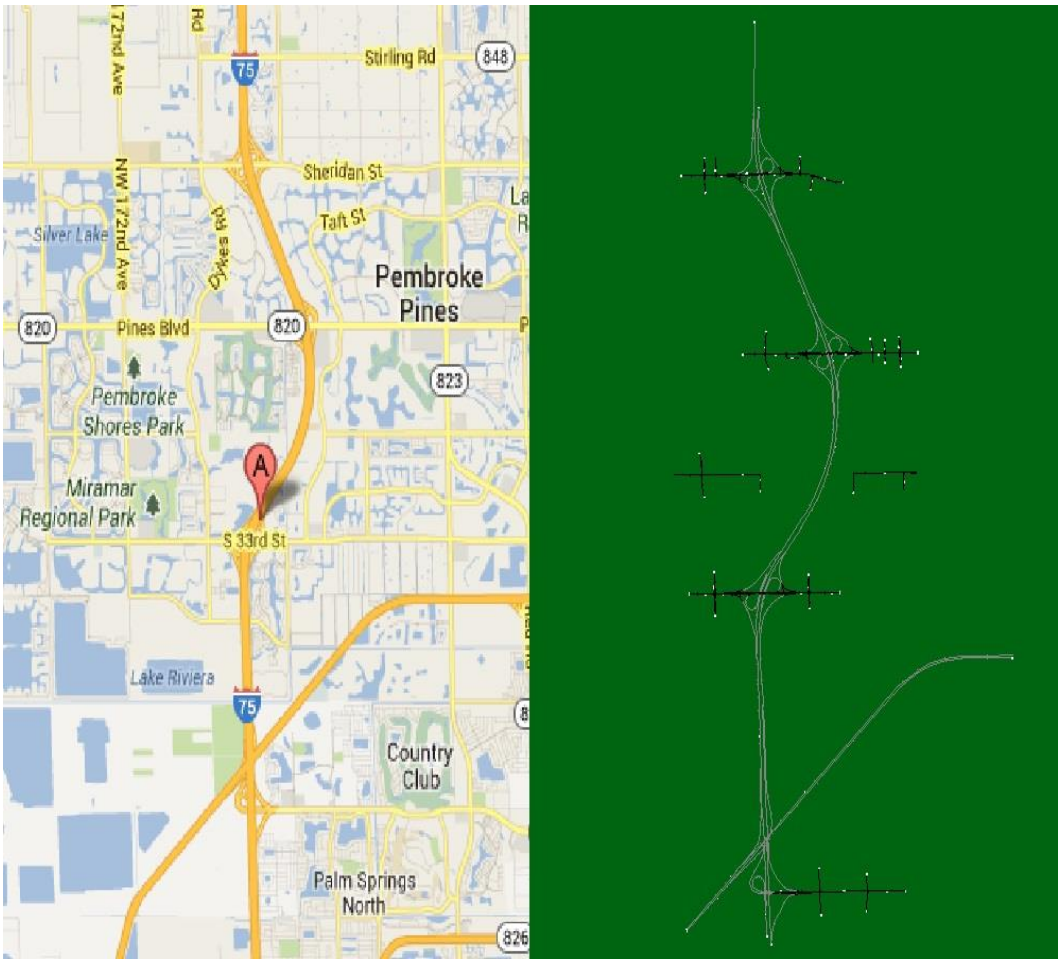
Figura 11. Autopista Pyramid, Reno, NV. (a) Google Map, (b) Modelo CORSIM

La Figura 11 muestra el modelo CORSIM (a) y el mapa de Google Maps (b) de la autopista Pyramid en Reno Nevada. Número de enlaces: 126.

La finalidad que tuvo la ejecución de este experimento fue la de recolectar datos que evidenciaran el comportamiento de las instancias propuestas (además del alg. GA y SPSA) al calibrar un modelo de complejidad media.

4.3.4 Experimento 3: Modelo complejidad alta: I-75 network

La Figura 12 muestra una porción de la autopista interestatal I-75 en Miami, Florida. Número de enlaces: 703.



(a) (b)

Figura 12. I-75, Miami, FL. *(a)* Mapa Google, *(b)* Modelo CORSIM.

La finalidad que tuvo la ejecución de este experimento fue la de recolectar datos que evidenciaran el comportamiento de las instancias propuestas (además del alg. GA y SPSA) al calibrar un modelo de complejidad alta.

4.4 EJECUCIÓN DEL EXPERIMENTO

4.4.1 Recursos

El prototipo software fue ejecutado en un servidor dedicado, provisto por el departamento de transporte de La Universidad de Nevada, Las Vegas, sus características son las siguientes:

Sistema operativo: Windows Server 2008
Procesador: Intel Xeon de 64 núcleos.

Ram: 256 Gb ddr3
Software: Java Virtual Machine V1.7

4.4.2 Ejecución de experimentos

Cada experimento fue ejecutado un total de 3 veces con el fin de verificar su correcto funcionamiento, es decir cada modelo fue calibrado 3 veces mediante la instancia del algoritmo memético con algoritmo genético y búsqueda tabú, la instancia del algoritmo memético con algoritmo genético y enfriamiento simulado, un algoritmo genético y el algoritmo SPSA.

El algoritmo SPSA [54] es un algoritmo de optimización estocástica, sobre el cual se utilizó una función de parada determinada por el GEH, medida que permite conocer si un modelo esta calibrado según la Federal Highway Administration, directriz que se utilizó en todos los algoritmos expuestos, además se limitó a un número máximo de 150 iteraciones.

4.5 AJUSTES DE LOS ALGORITMOS

Sobre cada algoritmo se realizó un proceso de afinación de parámetros con el fin de fijar los valores necesarios sobre los parámetros de los algoritmos para que estos pudieran funcionar como herramienta de calibración, los valores de los parámetros fueron modificados dentro de límites sugeridos por la documentación encontrada durante la revisión de la literatura, el número aproximado de experimentos realizados para que cada algoritmo pudiera funcionar de manera adecuada fue de 10, es decir que se realizaron alrededor de 40 experimentos para que los algoritmos funcionarán de manera correcta sobre el problema que presenta el realizar calibración sobre modelos de calibración. Durante el capítulo 5 se exponen los valores de los parámetros que se determinaron mediante los ajustes aquí mencionados sobre cada uno de los algoritmos.

4.6 EVALUACIÓN DE RESULTADOS Y COMPARATIVA

Cada modelo expuesto en el diseño de los experimentos fue calibrado mediante cuatro algoritmos diferentes, la instancia del algoritmo memético con algoritmo genético y búsqueda tabú, la instancia del algoritmo memético con algoritmo genético y enfriamiento simulado, un algoritmo genético y el algoritmo SPSA.

A partir de los resultados obtenidos de realizar las calibraciones mencionadas se evalúa si los algoritmos pudieron o no calibrar el modelo, además se obtienen medidas de tiempo y precisión de los algoritmos a partir de la medida de la media cuadrática normalizada, función que se utiliza como función de evaluación sobre los diferentes algoritmos

desarrollados. A partir de estas medidas se realizan tablas y graficas que permiten comparar el comportamiento de los diferentes algoritmos usados. La evaluación y comparativa de resultados se expone en el capítulo 5 del presente documento.

Capítulo 5

5. ALGORITMO MEMÉTICO PARA CALIBRACIÓN DE MODELOS DE MICROSIMULACIÓN DE FLUJO DE TRÁFICO VEHICULAR CORSIM

La instancia del algoritmo memético está compuesta por un algoritmo genético como algoritmo de exploración y un algoritmo de recocido simulado o de búsqueda tabú como algoritmo de explotación. Dado el gran tamaño de los modelos de micro simulación de flujo de tráfico vehicular, y de manera particular los modelos CORSIM, se debe garantizar una búsqueda extensiva (exploración) a través del vasto espacio de soluciones y una búsqueda intensiva (explotación) en las soluciones con mayor probabilidad de obtener el óptimo global. La Figura 13 ilustra el modelo de algoritmo memético usado, este modelo es propuesto en [55]

Algoritmo Memético Usado

Entrada: Modelo sin calibrar Θ; Parámetros de campo α
Salida: Modelo calibrado Θ_c

```
población ← GenerarPoblacionInicial
   mientras CriterioParada(Θ,α) hacer
      seleccionados ← Selección(población)
      cruzados ←Cruce(seleccionados)
      población ← Mutación(cruzados)
      OptimizaciónLocal(población)
      Remplazo(población)
   fin mientras
    Θc ← Mejor(población)
Retornar Θc
```

Figura 13. Algoritmo memético a usar para la calibración

Se define $\boldsymbol{\theta}$ como el conjunto de parámetros de entrada de la simulación, $\boldsymbol{\gamma}$ como el resultado o salida del proceso de simulación y $\boldsymbol{\alpha}$ como los datos de campo recolectados en entornos reales.

$$\theta = \{x_1, x_2, x_3, \dots, x_i, \dots, x_n\}$$
$$\gamma = \{y_1, y_2, y_3, \dots, y_i, \dots, y_m\}$$
$$\alpha = \{z_1, z_2, z_3, \dots, z_i, \dots, z_m\}$$

n es el tamaño de la solución.

m es el tamaño de las salidas de la simulación.

Por tanto θ representa una solución candidata, de aquí en adelante solución, al problema de calibración, el cual consiste en encontrar θ_c (modelo calibrado) tal que la salida de simular θ_c sea cercana a los datos medidos en campo, es decir:

$$\gamma = simulacion(\theta_c) \quad \rightarrow \quad \gamma \approx \alpha$$

Cada x_i representa un parámetro de entrada de la simulación que está sujeto a restricciones de dominio, es decir:

$$x_{i\,min} \leq x_i \leq x_{i\,max}$$

El algoritmo memético ilustrado por la Figura 13 se ejecuta de la siguiente manera: En la línea 1 se crea la población inicial, se crean 16 individuos aleatoriamente tomando como base la solución inicial, es decir el modelo sin calibrar. Mientras no se cumpla el criterio de parada se itera en las líneas 3 a 7, cada iteración obtiene una nueva generación de individuos, en esta sección se realiza el proceso de selección, cruce y mutación de la población previamente generada, en la línea 6 se realiza un proceso de mejora local sobre el mejor individuo de la población, esta mejorase hace usando un algoritmo de búsqueda tabú o recocido simulado. Al final se retorna el mejor individuo obtenido a través de las diferentes generaciones.

La elección de cómo aplicar el operador de optimización local se basó en la anatomía de un algoritmo genético planteada por [55], por tanto se optó por aplicar el optimizador local solamente al mejor individuo de cada generación, esto dado que la complejidad computacional de aplicarlo a un porcentaje de la población es bastante alta y el tiempo de ejecución crece linealmente de acuerdo al número de individuos a los que se les aplique mejora local; en lugar de ello se optó por aplicar el operador de mutación antes de la optimización local con el fin de garantizar la característica de inclusión de nuevo conocimiento del algoritmo memético.

5.1 DESCRIPCIÓN DE UN INDIVIDUO

La Figura 14 ilustra a manera de ejemplo la estructura y contenido de los archivos planos que utiliza el simulador CORSIM como entrada para el proceso de simulación, es decir el conjunto θ.

La información es abstraída de tal manera que los datos se encuentran consignados en grupos de columnas, a estos grupos de columnas se les denomina entries (entradas), cada línea representa un record type especifico por tanto cada entry almacena los datos propios de un record type.

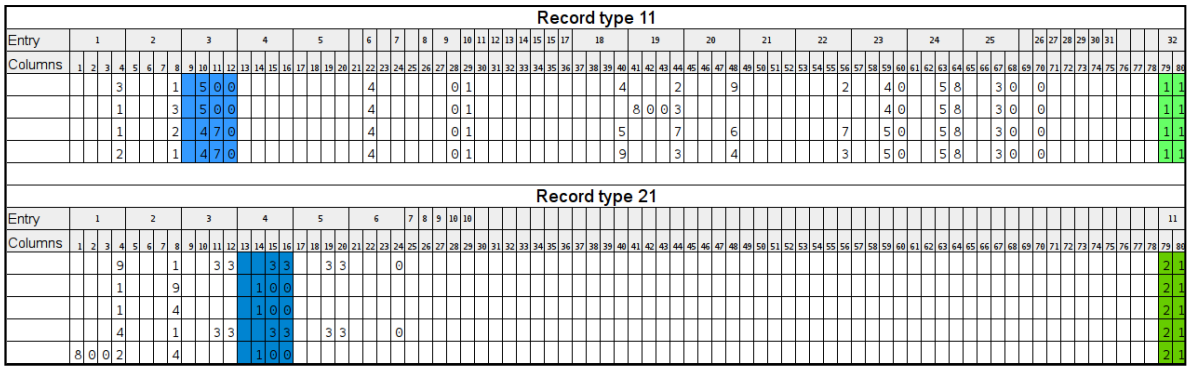
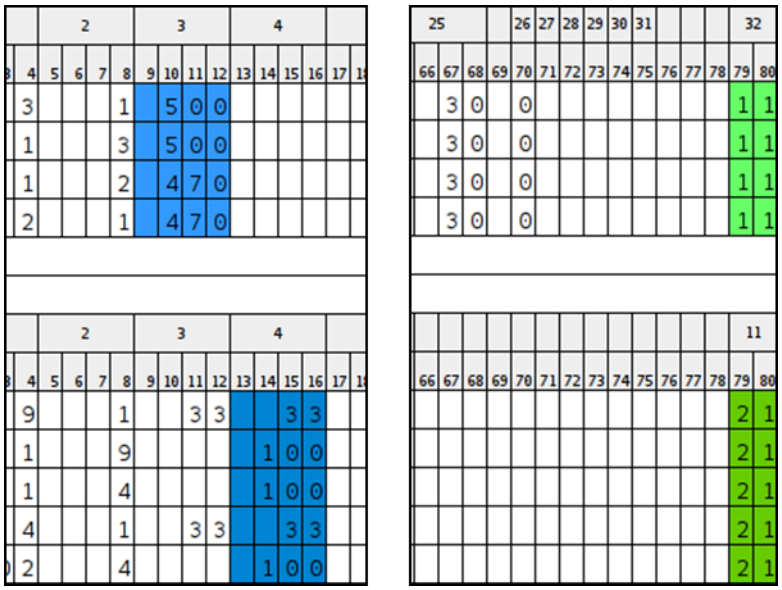


Figura 14. Estructura de archivos planos en formato de CORSIM para ser simulados

La Figura 15 hace énfasis en 2 partes de la figura 15, la parte (b) ilustra los record type 11 en verde claro y 21 en verde oscuro. El record type 11 contiene información sobre las características de una calle, estas características se almacenan en cada una de sus entries; el record type 21 contiene información sobre el volumen de vehículos que gira en una intersección.

La Figura 15 (a) representa resaltado en azul claro la entry 3 para el record type 11, esta entry almacena la longitud de una calle, adicionalmente resaltado en azul oscuro representa la entry 4 para el record type 21 que contiene información sobre al número de vehículos que giran diagonal en un determinado cruce. Estos valores resaltados en azul son los datos a calibrar.



(a) (b)

Figura 15. Ejemplo datos modelo CORSIM, (a) Entries con valores, (b) Record type

Dados los datos a calibrar resaltados en azul claro y oscuro, el individuo resultante será el ilustrado por la Figura 16.

500	500	470	470	33	100	100	33	100

Figura 16. Ejemplo individuo

De lo anterior es posible deducir como la longitud n de la solución depende del tamaño del modelo a calibrar y de los record type seleccionados con sus respectivas entries. Los valores que puede tomar cada x_i son enteros y a continuación se explican sus restricciones.

El manual de referencia CORSIM [4] explica de manera detallada el significado de cada uno de los record type con sus respectivas entries.

5.2 RESTRICCIONES SOBRE EL INDIVIDUO

Cada conjunto de parámetros agrupados por un record type pueden estar sujetos a 2 tipos de restricciones.

a) Los valores de los parámetros deben estar entre un máximo y un mínimo valor.

b) Los valores de los parámetros de un mismo record type deben conservar coherencia.

Por ejemplo, como se mencionó previamente, el record type 21 contiene 4 valores de parámetros que representan porcentajes de vehículos que giran a la derecha, a la izquierda, al frente y diagonal en un determinado cruce. Las restricciones para cada uno de estos 4 parámetros del record type 21 estarán dadas por, un mínimo valor de cero y un máximo valor de 100, además que la suma de los 4 valores debe dar como resultado 100 ya que se debe conservar la coherencia en estos valores de porcentaje.

Las restricciones para cada record type se encuentran en el manual de referencia CORSIM [4].

5.3 DESCRIPCIÓN DE UNA SOLUCIÓN CERCANA O INDIVIDUO CERCANO

El concepto de solución cercana o vecina es de gran relevancia, por tanto, se define una solución vecina θ_v como aquella solución a la que se puede llegar desde una solución original θ a través de cambios en su configuración.
A continuación se detalla paso a paso el proceso de generación de una solución vecina.

Dada la solución original θ

x_1	x_2	x_3	x_4	x_5	x_6	…	x_i	…	x_n

Se calcula Δ_{x_i}, diferencia entre los valores máximos y mínimo de cada parámetro x_i

$$\Delta_{x_i} = x_{i\,max} - x_{i\,min}$$

Δ_{x_1}	Δ_{x_2}	Δ_{x_3}	Δ_{x_4}	Δ_{x_5}	Δ_{x_6}	…	Δ_{x_i}	…	Δ_{x_n}

Se calcula la alteración ρ_i, como un 10% de Δ_{x_i}

$$\rho_i = \Delta_{x_i} * 0.1$$

ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6	…	ρ_i	…	ρ_n

Con una probabilidad del 50% se establecen alteraciones positivas o negativas y se aplicara la alteración solo al 30% de la longitud n de la solución.

$-\rho_1$		$+\rho_3$			$+\rho_6$	…	$-\rho_i$	…	$+\rho_n$

Finalmente se aplican estas alteraciones, dando como resultado la solución vecina θ_v

$x_1 - \rho_1$	x_2	$x_3 + \rho_3$	x_4	x_5	$x_6 + \rho_6$	…	$x_i - \rho_i$	…	$x_n + \rho_n$

La definición de los valores de: alteración del 10%, porcentaje de suma o resta del 50%, porcentaje de aplicación de la alteración del 30%, se realizó de manera experimental; se hizo una variación en los porcentajes y se analizaron los resultados del algoritmo.

5.4 FUNCIÓN OBJETIVO

La función objetivo previamente definida como *la media cuadrática normalizada* [56] esta es expresada matemáticamente como lo ilustra la ecuación (1).

$$f(\gamma,\alpha) = NRMS = \frac{1}{\sqrt{n}} * \left(W * \sqrt{\sum_{i=1}^{n} \left(\frac{\alpha_{v,i,t} - \gamma_{v,i,t}}{\gamma_{v,i,t}} \right)^2} + (1 - W) * \sqrt{\sum_{i=1}^{n} \left(\frac{\alpha_{s,i,t} - \gamma_{s,i,t}}{\gamma_{s,i,t}} \right)^2} \right) \quad (1)$$

Donde:
$\alpha_{v,i,t}$ = datos de campo de volumen vehicular para el enlace *i* en el tiempo *t*
$\gamma_{v,i,t}$ = salida de la simulación de volumen vehicular para el enlace *i* en el tiempo *t*
$\alpha_{s,i,t}$= datos de campo de velocidad para el enlace *i* en el tiempo *t*
$\gamma_{s,i,t}$ = salida de la simulación de velocidad para el enlace *i* en el tiempo *t*
W = peso usado para asignar mayor o menor ponderación a los volúmenes o velocidades

El objetivo del algoritmo es que la diferencia entre los datos de salida de la simulación γ y los datos de campo α sean lo mas cercanos posible, por tanto el propósito del algoritmo es minimizar la función objetivo dada.
De acuerdo a la fiabilidad de los datos se le da mayor o menor peso a las medidas de volumen o velocidad, por ejemplo, si para determinada calibración se conoce que los datos de velocidad se midieron con mayor precisión entonces se debe dar un valor bajo a W de tal manera que se dé más ponderación a los datos de velocidad.

5.5 OPERADORES DE SELECCIÓN, MUTACIÓN Y CRUCE

La configuración o instancia del algoritmo genético usada fue determinada de manera experimental al analizar los resultados de varias corridas del algoritmo.
La selección se realiza a través torneo; aleatoriamente se eligen dos individuos de la población, con una probabilidad de 0.7 se elige el mejor de los individuos y con una probabilidad del 0.3 se elige el peor, así se obtiene el primer padre y se repite el proceso para obtener el segundo, este método de selección representa 2 grandes ventajas, primero respecto a rendimiento y segundo en cuanto a facilidad para programación en paralelo, estas 2 son características deseables del algoritmo de calibración, además ha sido exitosamente usado en investigaciones previas [57].

Los padres generados previamente se cruzan usando un método de cruce uniforme basado en una máscara, es decir se crea un vector del mismo tamaño del individuo θ, llamaremos θ_A al primer padre y θ_B al segundo.
Siendo θ_Ay θ_B respectivamente

x_{A1}	x_{A2}	x_{A3}	x_{A4}	x_{A5}	x_{A6}	$\cdots$	x_{Ai}	$\cdots$	x_{An}

x_{B1}	x_{B2}	x_{B3}	x_{B4}	x_{B5}	x_{B6}	$\cdots$	x_{Bi}	$\cdots$	x_{Bn}

Y una máscara aleatoriamente generada:

0	0	1	0	1	1	…	1	…	0

Dada la máscara anterior se cruzan solamente los elementos con valor 1, el resultado del cruce será θ_Ay θ_B respectivamente:

x_{A1}	x_{A2}	x_{B3}	x_{A4}	x_{B5}	x_{B6}	$\cdots$	x_{Bi}	$\cdots$	x_{An}

x_{B1}	x_{B2}	x_{A3}	x_{B4}	x_{A5}	x_{A6}	$\cdots$	x_{Ai}	$\cdots$	x_{An}

Estos padres se cruzan con una probabilidad del 75%.
Experimentalmente se estableció este porcentaje de cruce, acotando su rango a partir de 45% a 90%.

Los hijos θ_Ay θ_B resultantes se mutan con una probabilidad del 7%, esta mutación se realiza usando el mismo criterio de solución cercana θ_v previamente definido.
A través de las generaciones se van generando nuevos individuos con mejor función objetivo, estos son agregados a la población y de acuerdo a la cantidad de individuos nuevos agregada se eliminan los individuos con peor función de evaluación conservando así el tamaño de la población.

5.6 CONDICIÓN DE PARADA

Las directrices provistas por la Federal Highway Administration (FHWA) para modelos CORSIM son aquí usadas. La estadística GEH [6] debe ser menor que 5 para al menos el 85% de los enlaces, pero añade que es la condición mínima, es decir que es un modelo aceptablemente calibrado, por tanto se estableció un GEH menor a 5 para al menos 95% de los enlaces. La estadística GEH se calcula como lo ilustra la ecuación (2).

$$GEH = \sqrt{\frac{2(\alpha_{v,i,t} - \gamma_{v,i,t})^2}{\alpha_{v,i,t} + \gamma_{v,i,t}}} \qquad (2)$$

$\alpha_{v,i,t}$ = datos de campo de volumen vehicular para el enlace *i*

$\gamma_{v,i,t}$= salida de la simulación de volumen vehicular para el enlace *i*

Adicionalmente se definen 100 iteraciones máximo para todo el algoritmo y 20 iteraciones máximo cuando no se encuentra mejora en la función de objetivo. Estos valores fueron definidos después de analizar los resultados de varias corridas del algoritmo, al notar un número de iteraciones donde no existía mejora.

5.7 TAMAÑO DE LA POBLACION INICIAL

Previas investigaciones [55] sugieren el tamaño de la población inicial de acuerdo a la longitud del individuo, o longitud del vector de solución θ, sin embargo, en el caso de los modelos de simulación la longitud de un individuo puede superar valores de miles, adicionalmente la función de evaluación implica correr una simulación completa y al tener un tamaño de población muy grande el proceso se torna lento, por tanto después de analizar el rendimiento al correr el simulador para evaluar un individuo se estableció un tamaño fijo de 16 en la población inicial.

5.8 ALGORITMOS DE OPTIMIZACIÓN LOCAL

5.8.1 Algoritmo de recocido simulado

La Figura 17 ilustra el algoritmo de recocido simulado usado en la presente investigación, este se basa en el algoritmo usado en [58]. El algoritmo itera las líneas 4 a 10 mientras no se cumpla con el criterio de parada. En la línea 4 se genera una solución vecina, esta usa el concepto de solución cercana θ_v previamente definido. Si la solución vecina tiene una mejor función objetivo que la mejor solución se actualiza como mejor solución, de lo contrario se evalúa la probabilidad de aceptación

Algoritmo Recocido Simulado

Entrada: Modelo para aplicar mejora local Θ; Parámetros de campo α
Salida: Modelo con mejora local Θ_c

```
1:  Θc ← Θ
2:    mientras CriterioParada(Θ , α) hacer
3:      Θ = GenerarSolucionVecina(Θ)
4:      si evaluacion(Θ) < evaluación(Θc) entonces
5:        Θc ← Θ
6:      sino
7:        ProbabilidadAceptacion(Θ , Θc)
8:      fin si
9:    fin mientras
10: Retornar Θc
```

Figura 17. Algoritmo de recocido simulado usado

5.8.1.1 Probabilidad de aceptación

La probabilidad de aceptación está dada por la ecuación (3).

$$e^{\left(\frac{NRMS_i - NRMS_{i-1}}{\tau_i}\right)} \quad (3)$$

Siendo τ_i la temperatura en la iteración i.

Esta probabilidad retorna un valor entre 0 y 1, por tanto, aleatoriamente se elige un número y si es menor que la probabilidad de aceptación calculada se acepta la nueva solución como solución actual del algoritmo.

5.8.1.2 Agenda de enfriamiento

Se utiliza un enfriamiento con función geométrica, este está representado por la ecuación (4) donde Siendo τ_i la temperatura en la iteración i, y τ_0 es la temperatura inicial.

$$\tau_i = 0.93\ \tau_0$$

El valor del coeficiente 0.93 se definió después de analizar la agenda de enfriamiento resultante de variar el coeficiente desde 0.85 a 0.98, tomando como referencia [59].

5.8.2 Algoritmo de búsqueda tabú

Algoritmo búsqueda tabú

Entrada: Modelo para aplicar mejora local Θ; Parametros de campo α
Salida: Modelo con mejora local Θ_c

```
1:  Θc ← Θ
2:    mientras CriterioParada(Θ , α) hacer
3:      vecindario = GenerarVecindario(Θ)
4:      para cada Θv de vecindario hacer
5:        si (evaluación(Θv ) < evaluación(Θc) y (NoEsTabu(Θv )) entonces
6:          Θc ← Θv
            HacerTabu(Θv)
7:        fin si
7:        DecrementarTabu()
8:      fin para
9:    fin mientras
10: Retornar Θc
```

Figura 18. Algoritmo de búsqueda tabú usado.

5.8.2.1 Generación de vecindario

Se estableció el tamaño del vecindario de 5 elementos, esto debido a la complejidad computacional que involucra evaluar cada individuo, al analizar el tiempo de ejecución después de varias corridas del algoritmo se analizó que el tamaño de vecindario mencionado daba un equilibrio entre rendimiento y precisión.

5.8.2.2 Lista tabú

La lista tabú se estableció con un tamaño de 10 basado en recomendaciones de [60] donde se estudia el comportamiento de la función objetivo al variar el tamaño de la lista tabú, se experimentó con valores diferentes para lista tabú y 10 fue el que mejor comportamiento evidencio en los resultados del algoritmo.

Capítulo 6

6. EXPERIMENTACIÓN Y RESULTADOS

A continuación se exponen los resultados obtenidos de los experimentos de diferente complejidad aquí realizados, para los datos de campo provistos se recomendó por parte de los expertos en tráfico de NDOT que el peso W de los datos de volumen en la función objetivo fuera de 0.7 (70%) y de 0.3 (30%) para los datos de velocidad, esto dado a que los datos de velocidad afectan a pocos de los parámetros de los modelos, además los datos de volumen ayudan a determinar en mayor medida las zonas de embotellamiento en un modelo, información de gran relevancia para Ingenieros de tráfico y afines.

6.1 MODELO COMPLEJIDAD BAJA: MAC TRANS NETWORK

Este modelo incluye 20 enlaces y se posee datos de campo para cada uno de ellos; la longitud de una solución o individuo para este modelo es de 42.

6.1.1 Resultados con SPSA

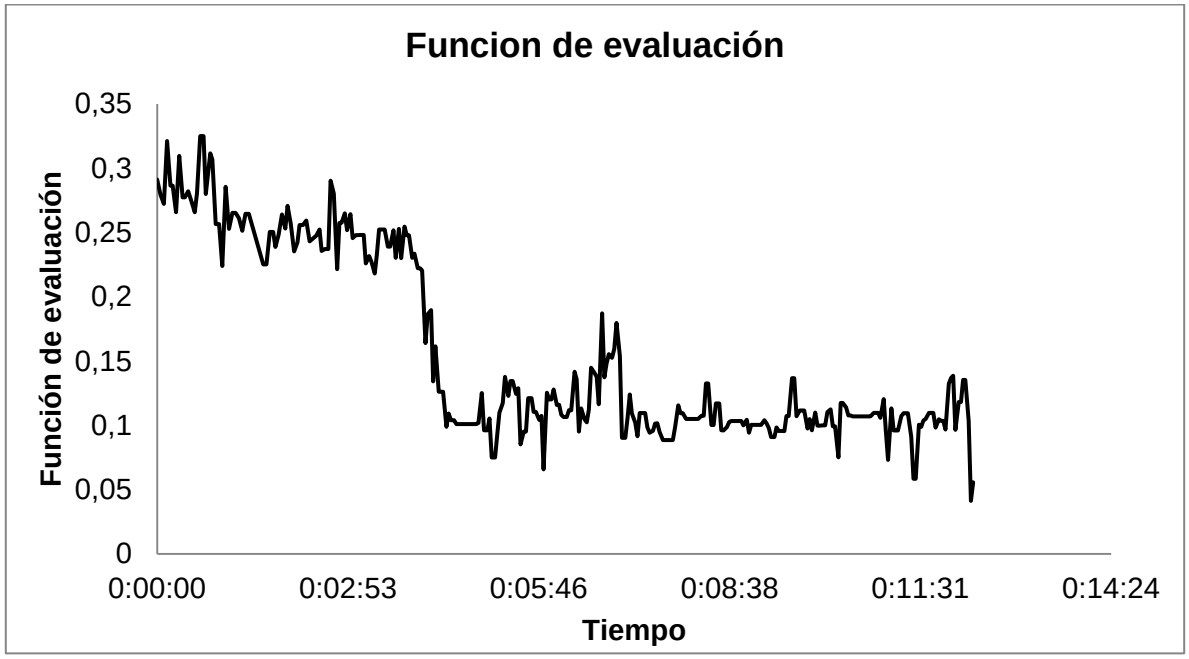


Figura 19. Función objetivo, SPSA, Mac Trans

La Figura 19 ilustra el comportamiento de la función objetivo para el algoritmo SPSA, dado la naturaleza del SPSA de algoritmo de optimización estocástico, en ocasiones la función objetivo presenta desmejoras pero la tendencia es hacia la convergencia. Este algoritmo alcanza una mejora desde 0,291182514 hasta 0,0584579. Este algoritmo se detiene dado el mismo criterio de parada, basado en la estadística GEH, esto es evidente en la Figura 20 donde se ilustra como la estadística GEH es menor a 5 para el 100% de los enlaces.

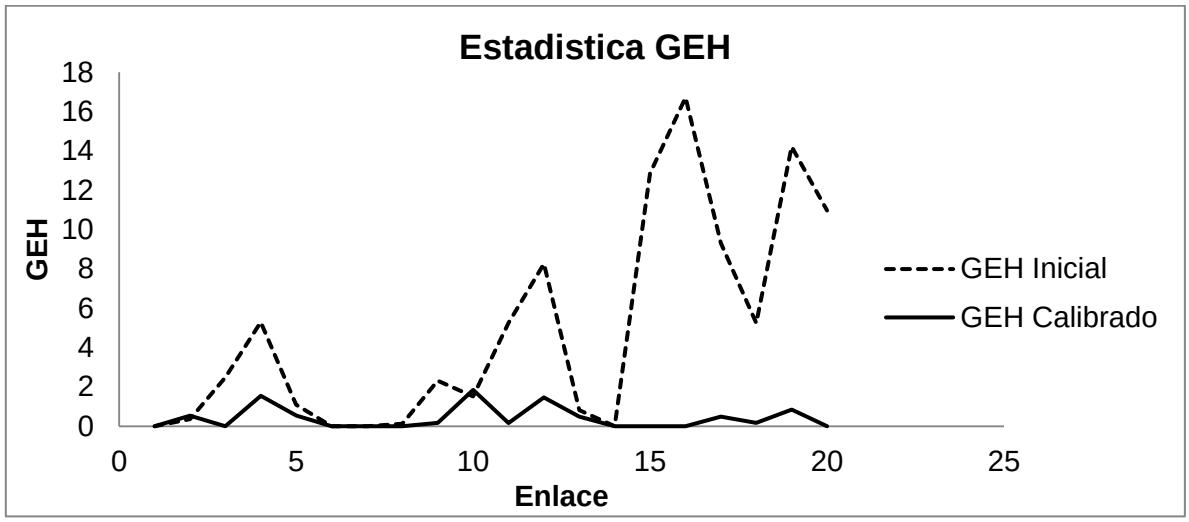


Figura 20. Estadística GEH, SPSA, Mac Trans

6.1.2 Resultados con instancia de algoritmo genético sin optimizador local.

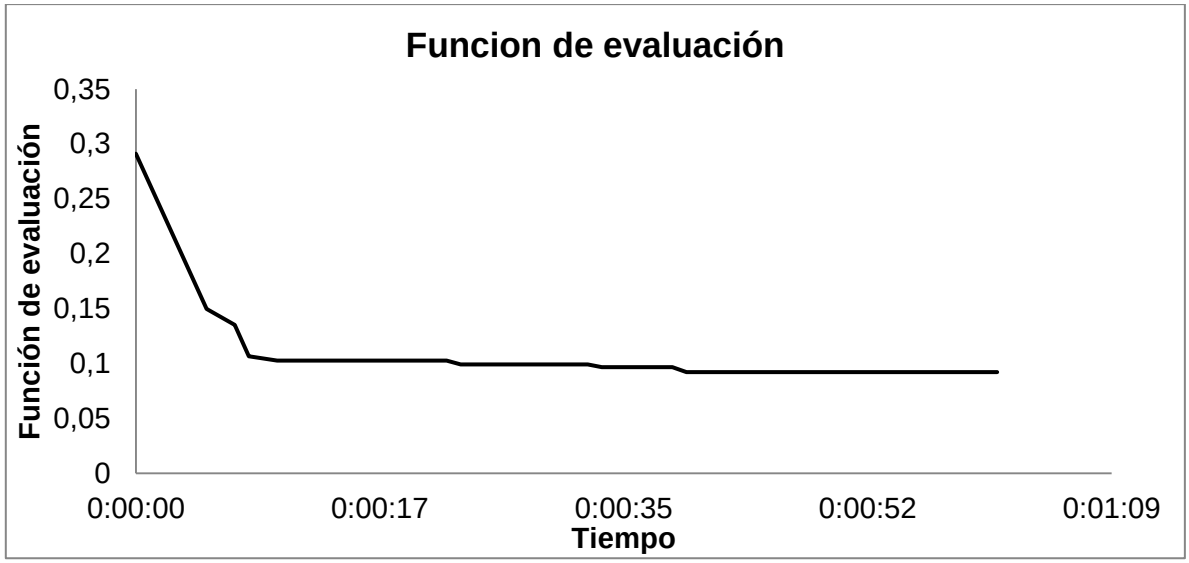


Figura 21. Función de objetivo, instancia con algoritmo genético sin optimizador local, Mac Trans

La Figura 21 ilustra el comportamiento de la función objetivo a través del tiempo, es evidente como el algoritmo se detiene después de un determinado tiempo, esto es debido a que el criterio de parado basado en el GEH no alcanza a cumplirse por tanto se detiene dado el segundo criterio del número máximo de iteraciones. Sin embargo es notoria una mejora en su función de evaluación al pasar de 0,291182514 a 0,092211153.

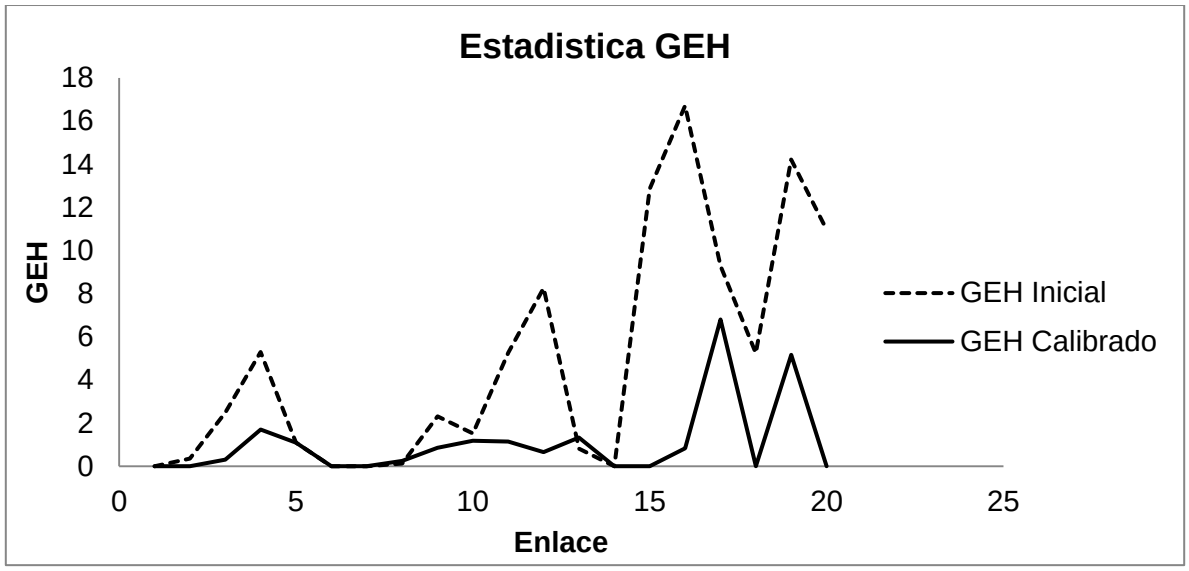


Figura 22. Estadística GEH, instancia con algoritmo genético sin optimizador local, Mac Trans

La Figura 22 ilustra la mejora obtenida en la estadística GEH para cada enlace; para el modelo inicial representado por la línea punteada la estadística GEH era superior a 5 para la mayoría de los enlaces y después de la calibración la estadística era superior a 5 solo para dos de los enlaces, sin embargo no es suficiente para considerar el modelo calibrado.

6.1.3 Resultados con instancia de algoritmo genético más recocido simulado

La Figura 23 ilustra el comportamiento de la función objetivo a través del tiempo, es posible notar como el algoritmo no se detiene después de un determinado tiempo sino en una mejora determinada lo que quiere decir que se ha cumplido el criterio de parada a través de la estadística GEH, lo cual se puede evidenciar en la Figura 24 en donde es notorio que para el GEH calibrado ninguno de los enlaces tiene un valor superior a 5 lo que evidencia el cumplimiento del criterio de parada. El GEH inicial era menor a 5 para el 55% de los enlaces, pero después de la calibración el GEH es menor a 5 para el 100% de los enlaces. La mejora en la función objetivo se da desde 0,291182514 hasta 0,024645163

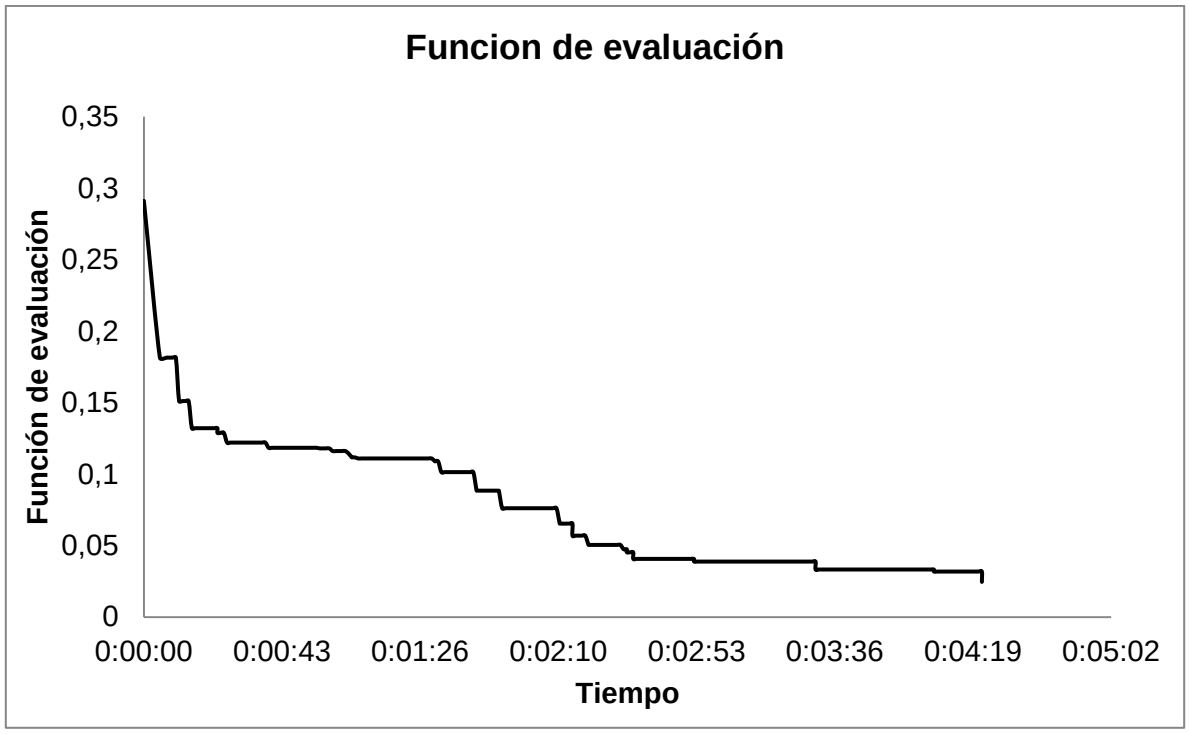


Figura 23. Función objetivo, instancia con algoritmo genético más recocido simulado, Mac Trans

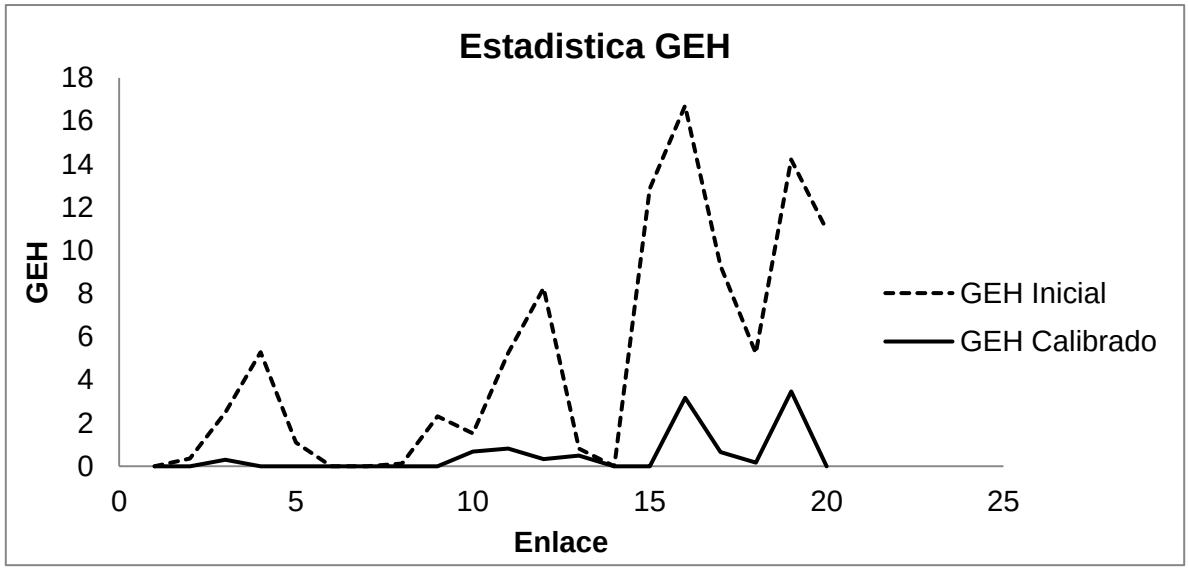


Figura 24. Estadística GEH, instancia con algoritmo genético más recocido simulado, Mac Trans

6.1.4 Resultados con instancia de algoritmo genético más búsqueda tabú

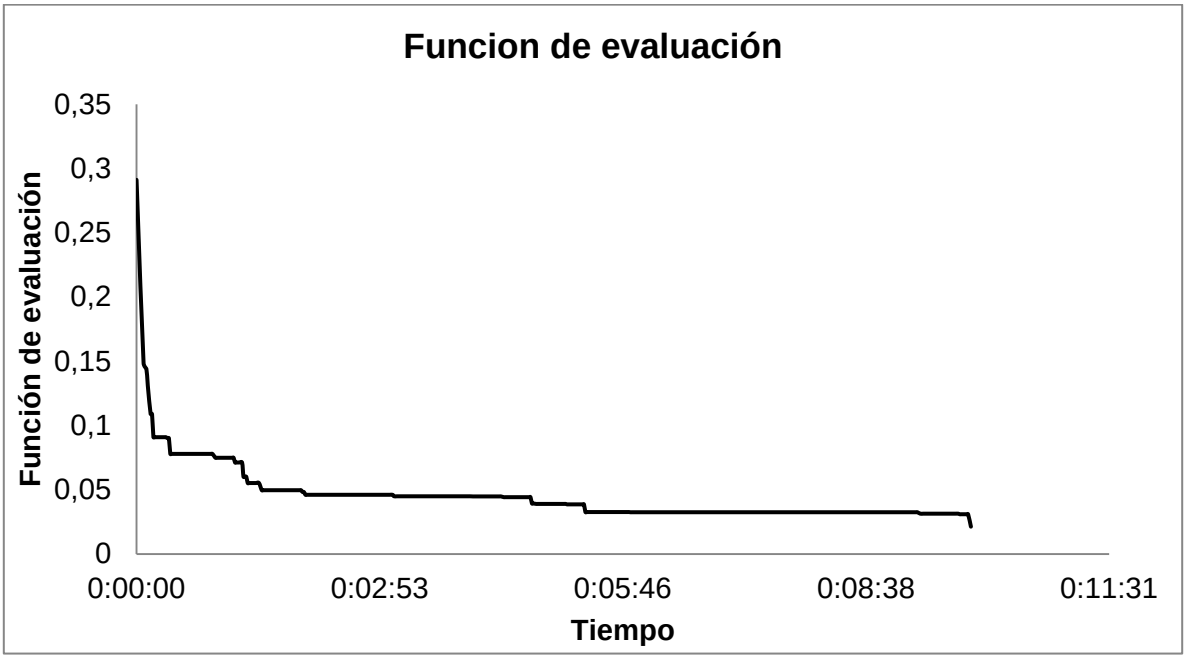


Figura 25. Función objetivo, instancia con algoritmo genético más búsqueda tabú, Mac Trans

La Figura 25 ilustra el comportamiento de la función objetivo a través del tiempo, al igual que la anterior grafica de función objetivo se puede notar como el algoritmo se detiene al alcanzar determinada mejora, lo cual quiere decir que el criterio de parada dado por la estadística GEH es cumplido, la mejora de función objetivo es desde 0,291182514 hasta 0,021343243 y el algoritmo se detiene a la iteración 23.

La Figura 26 ilustra la mejora obtenida en la estadística GEH para cada enlace, al igual que la gráfica anterior de estadística GEH es notorio como después de la calibración el GEH es menor que 5 para el 100% de los enlaces lo cual da evidencia de que el algoritmo se detuvo dado el criterio de parada que involucra esta estadística.

El comportamiento observado en las gráficas permite concluir la efectividad del método de calibración.

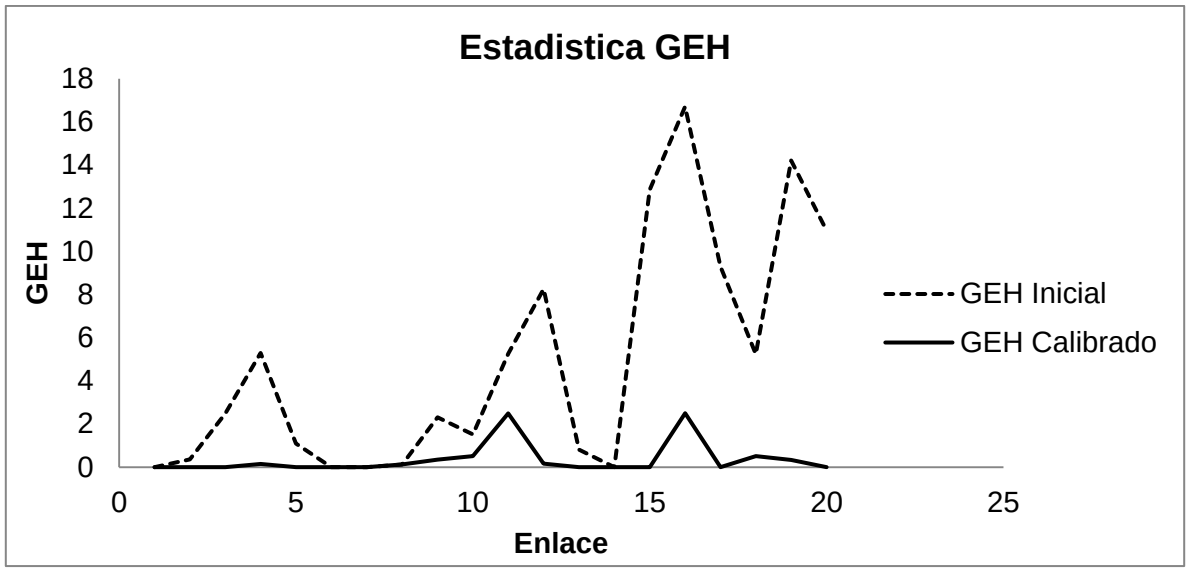


Figura 26. Estadística GEH, instancia con algoritmo genético más búsqueda tabú, Mac Trans

6.1.5 Comparación de los algoritmos

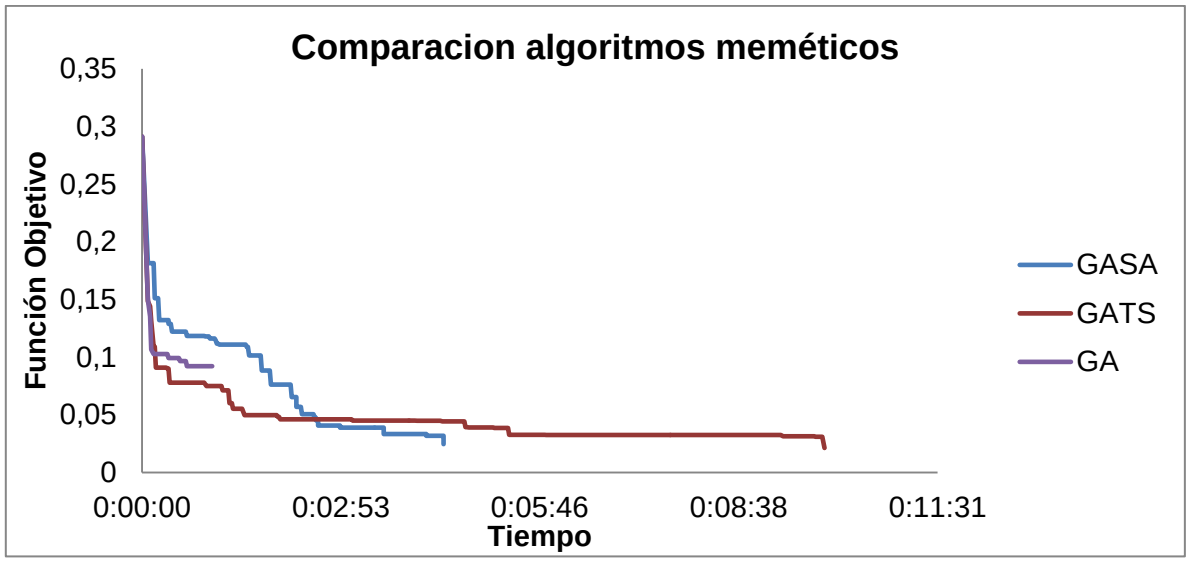


Figura 27. Comparación función objetivo algoritmos meméticos usados para calibración

La Figura 27 ilustra el comportamiento de la función objetivo para cada una de las 3 instancias de algoritmo memético usadas para calibrar la red MacTrans, es evidente por esta figura y por las figuras Figura 22, Figura 24 y Figura 26 como los 3 algoritmos logran resultados favorables en la calibración, sin embargo el algoritmo memético instanciado a través de un algoritmo genético y uno de búsqueda tabú (GATS) es el que obtiene un mejor resultado en la función objetivo. En cuanto al algoritmo memético sin optimizador

local, que aquí llamamos algoritmo genético (GA), se estima no alcanzo un valor más bajo en la función objetivo dado que no posee características significativas de explotación.

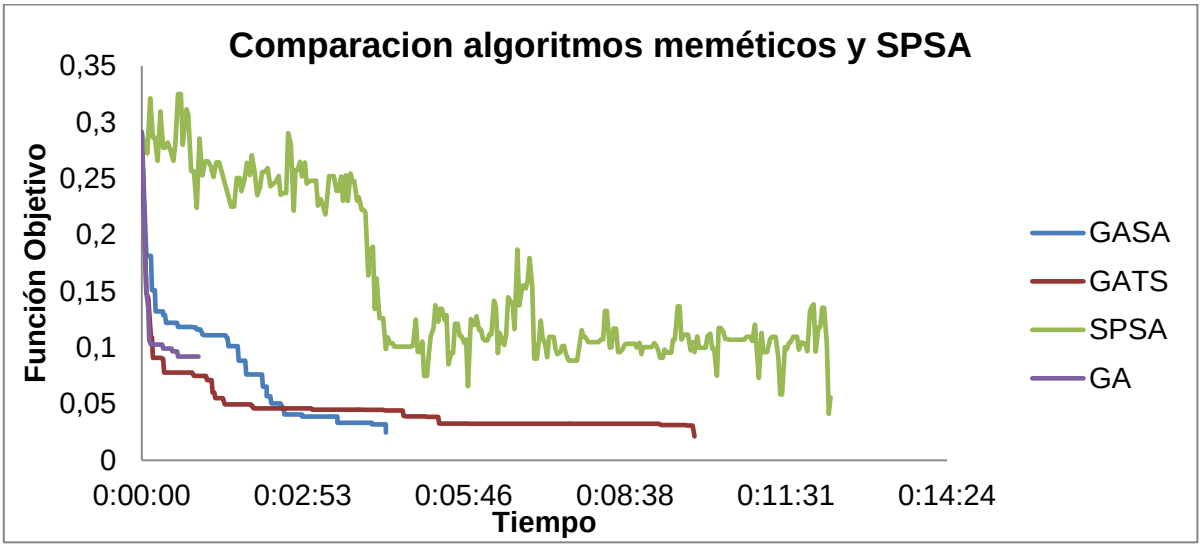


Figura 28. Comparación función objetivo de algoritmos meméticos y SPSA para calibración

La Figura 28 es una extensión de la Figura 27 añadiendo el comportamiento te la función objetivo del algoritmo SPSA la cual ilustra como el algoritmo con mejor comportamiento en cuanto a función objetivo es el algoritmo GATS y el algoritmo con menor número de iteraciones es el GASA, también es notorio como el SPSA tiene en cuanto a medida de la función objetivo tiene un comportamiento mejor que el GA, pero no mejor que GASA y GATS.

Tabla 1. Comparativa algoritmos de calibración, modelo Mac Trans

	GA	GASA	GATS	SPSA
NRMS Mínimo	0,09221115	0,02464516	0,0213432	0,0584579
Iteraciones	41	13	23	148
Tiempo Total	00:01:01	00:04:22	00:09:53	00:12:19
GEH	95%	100%	100%	100%

La Tabla 1 evidencia que en cuanto función objetivo la mejor instancia del algoritmo memético es GATS y en cuento a tiempo es GA, sin embargo GASA logra un resultado similar en función objetivo a GATS, pero GASA lo logra en un número de iteraciones menor y un tiempo menor. Por tanto de manera general se puede concluir que los algoritmo GASA y GATS evidenciaron un mejor comportamiento para calibrar la red Mac Trans.

6.2 MODELO COMPLEJIDAD MEDIA: RENO NETWORK

Esta red contiene 126 enlaces de los cuales se tiene datos de campo para 45; la longitud del individuo o solución para este modelo es de 288.

6.2.1 Resultados con SPSA

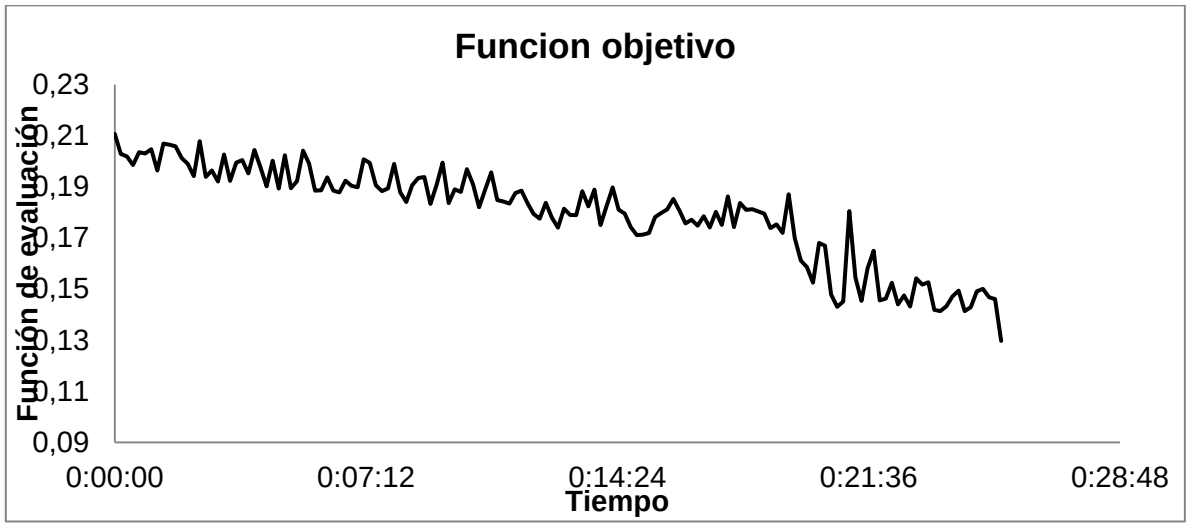


Figura 29. Función evaluación, SPSA, Reno

La Figura 29 ilustra la tendencia de la función objetivo hacia una convergencia, sin embargo el algoritmo se detiene después del número máximo de iteraciones, 150, definido para este ya que después de este número de iteraciones no logra mejora alguna en la función objetivo.

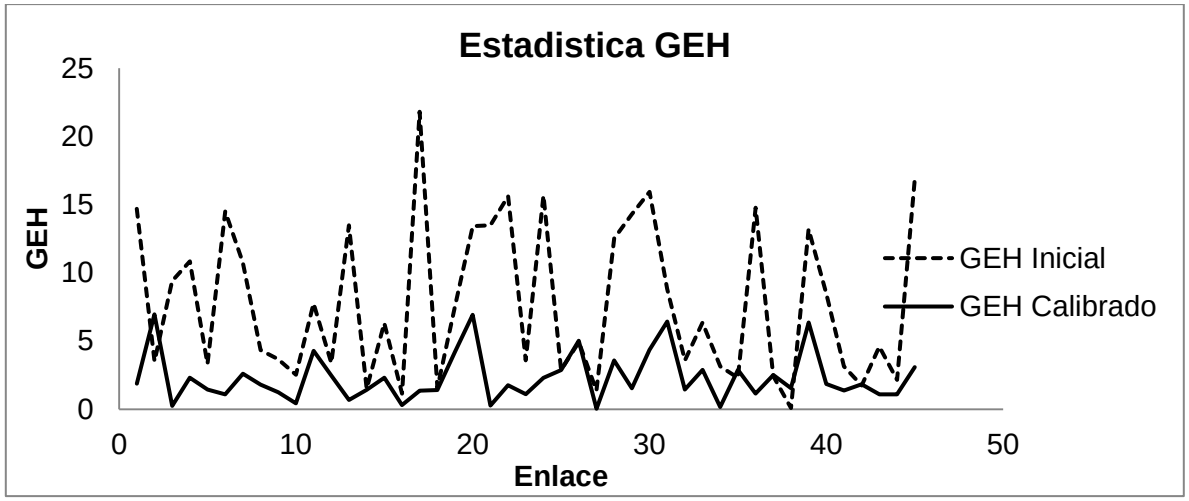


Figura 30. Estadística GEH, SPSA, Reno

La Figura 30 evidencia como el criterio de parada dado por la estadística GEH no es alcanzado por lo que se debe utilizar el segundo criterio de parada, la estadística GEH Inicial fue de menor a 5 para el 48,8% de los enlaces y después de la calibración fue de menor a 5 para el 88,8% de los enlaces lo cual no es suficiente para el criterio establecido.

6.2.2 Resultados con instancia de algoritmo genético sin optimizador local

La Figura 31 ilustra la tendencia a reducir la función objetivo de este algoritmo, se alcanza una mejora desde 0,210742987 hasta 0,143367881 y es posible notar como el algoritmo se detiene después de un tiempo determinado, esto dado que no se cumple el criterio de parada dado por la estadística GEH.

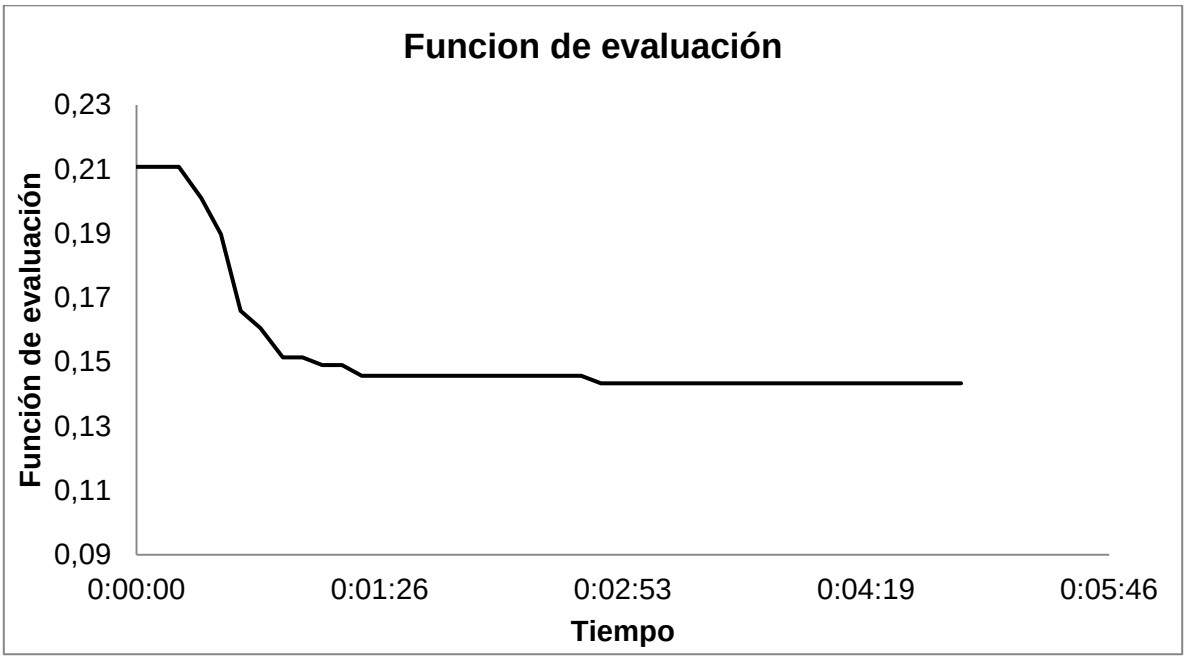


Figura 31. Función objetivo, instancia con algoritmo genético sin optimizador local, Reno

La afirmación anterior está respaldada por la Figura 32 donde se nota como varios enlaces tienen una estadística de GEH mayor a 5, más exactamente el 77,7% de los enlaces posee una estadística GEH menor a 5 lo cual evidencia un resultado no favorable en la calibración ya que no cumple ni con el criterio mínimo establecido por la FHWA de un 85%.

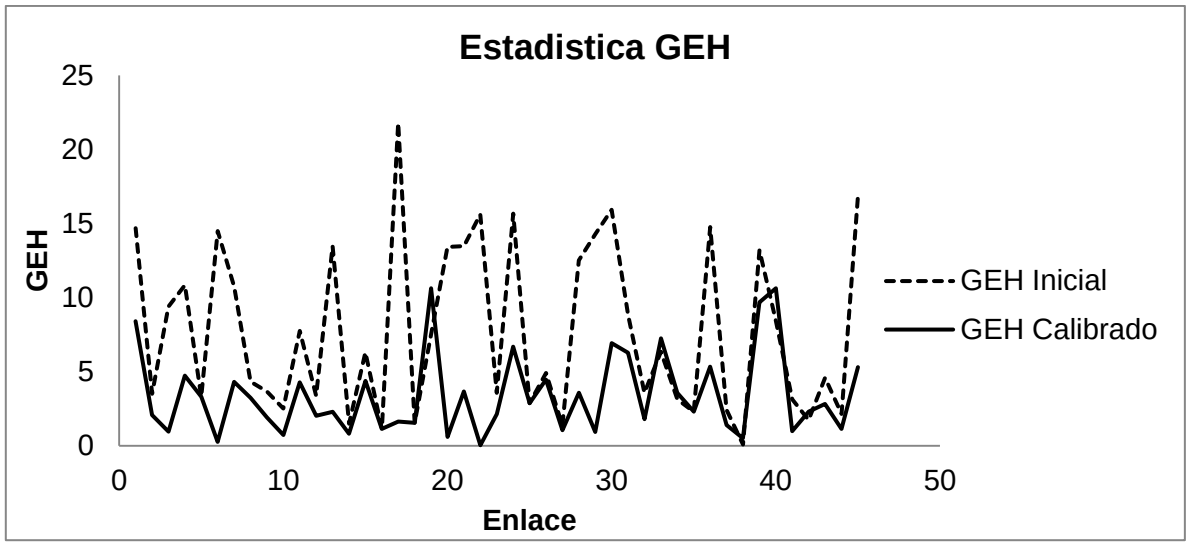


Figura 32. Estadística GEH, instancia con algoritmo genético sin optimizador local, Reno

6.2.3 Resultados con instancia de algoritmo genético más recocido simulado

El comportamiento de la función objetivo ilustrado por la Figura 33 ilustra la disminución de la función objetivo desde un valor de 0,210742987 hasta 0,113644539, adicionalmente es posible notar como el algoritmo se detiene por el criterio de un máximo número de iteraciones sin obtener mejora alguna sin cumplir así el criterio de la estadística GEH mínima aquí definida pero si cumpliendo el establecido por el FHWA. La Figura 34 ilustra cómo después de la calibración el GEH era menor a 5 para el 93,3% de los enlaces, sin alcanzar a cumplir el mínimo aquí establecido pero si cumpliendo el definido por la FHWA de 85%.

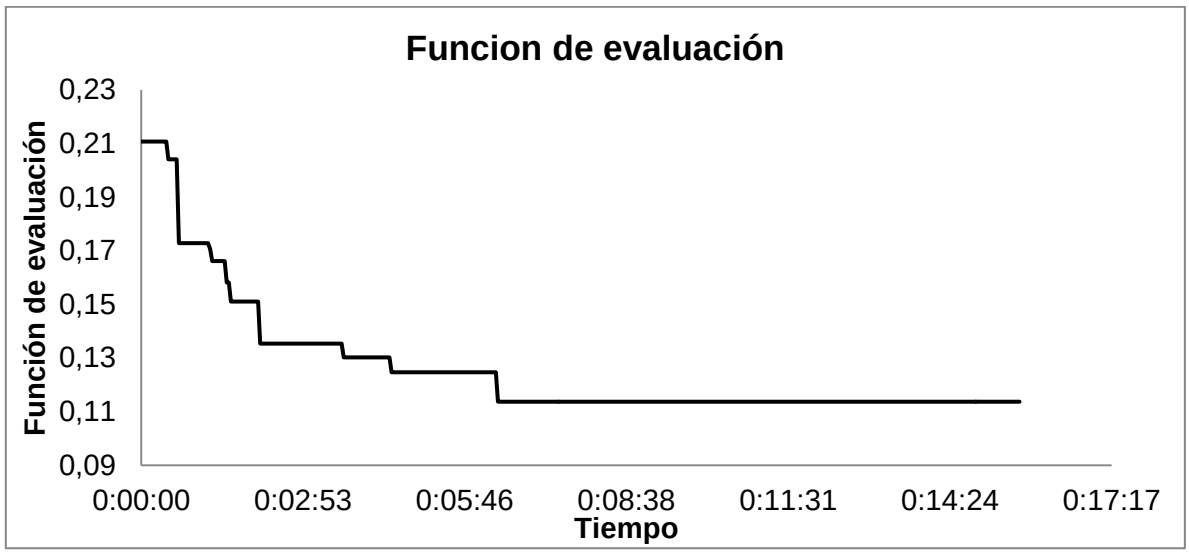


Figura 33. Función objetivo, instancia con algoritmo genético más recocido simulado, Reno

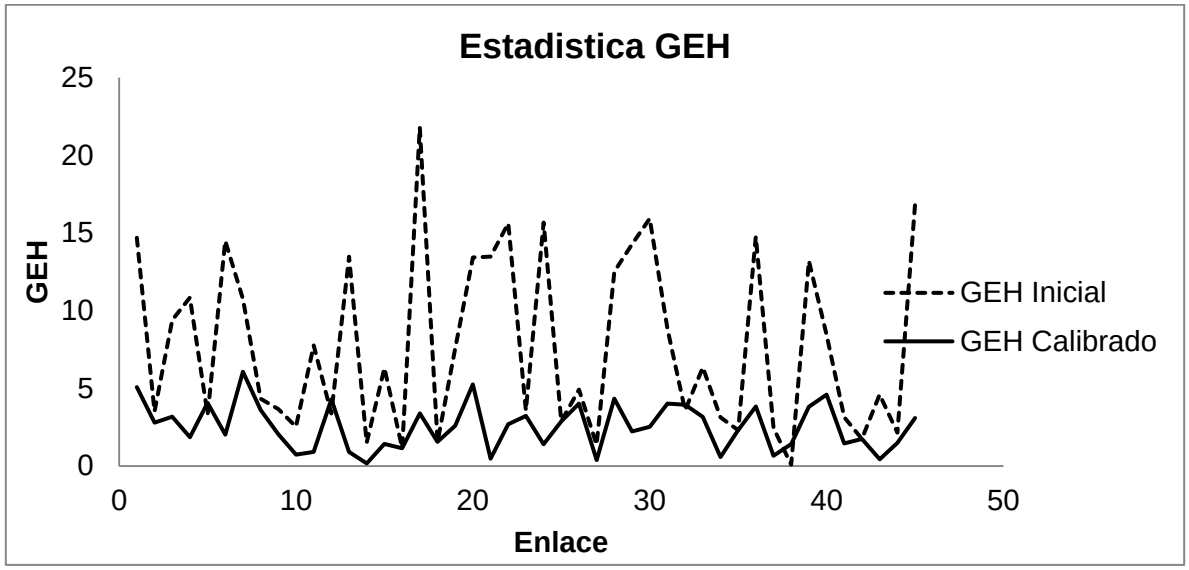


Figura 34. Estadística GEH, instancia con algoritmo genético más recocido simulado, Reno

6.2.4 Resultados con instancia de algoritmo genético más búsqueda tabú

La trayectoria de la función objetivo ilustrada por la Figura 35 evidencia una mejora desde 0,210742987 hasta 0,11625052, similar a la obtenida por el GASA para este mismo modelo, incluso similar en criterio de parada dado por el máximo número de iteraciones sin obtener mejora alguna en la función objetivo.

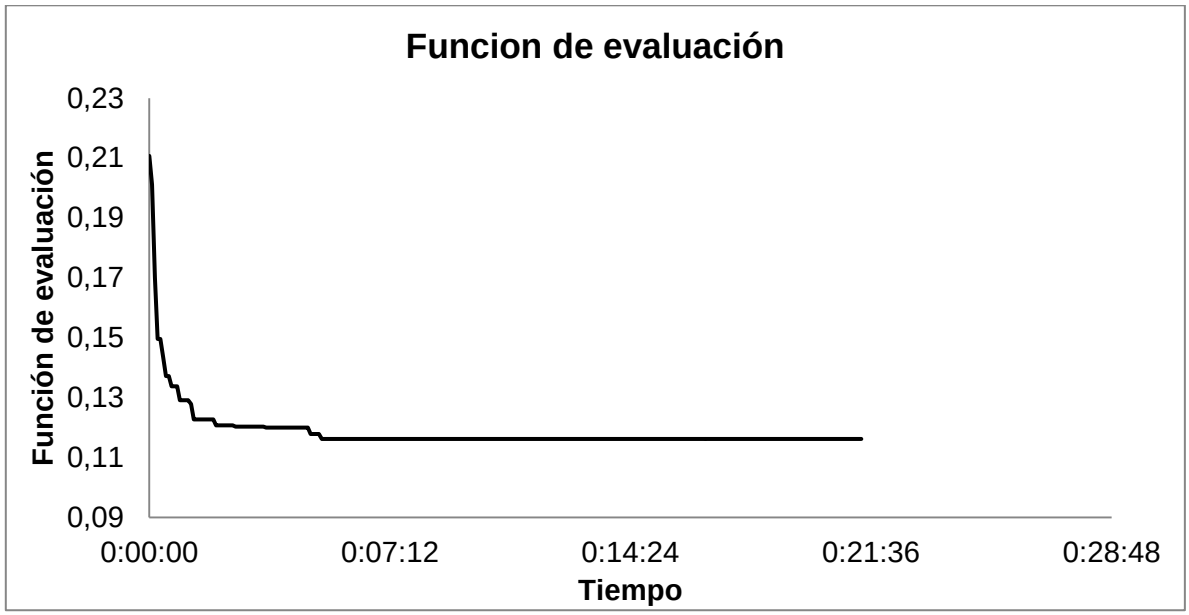


Figura 35. Función objetivo, instancia con algoritmo genético más búsqueda tabú, Reno

La estadística GEH ilustrada por la Figura 36 evidencia para después de la calibración un valore de GEH menor a 5 para el 93,3% de los enlaces lo cual no cumple el criterio de GEH mínimo aquí establecido pero si el definido por la agencia FHWA.

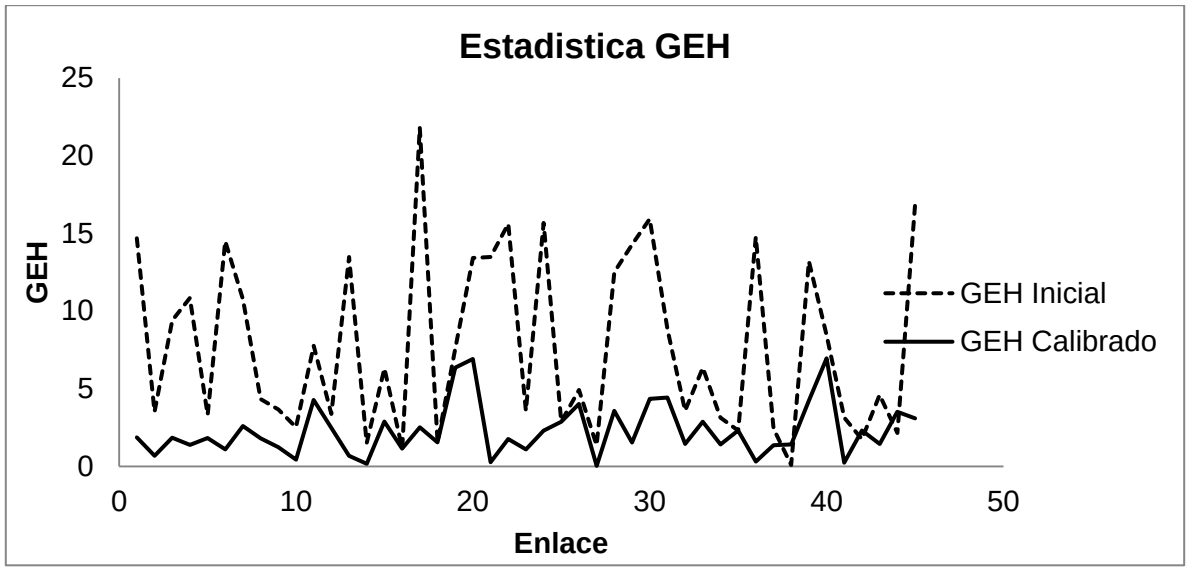


Figura 36. Estadística GEH, instancia con algoritmo genético más búsqueda tabú, Reno

6.2.5 Comparación de algoritmos

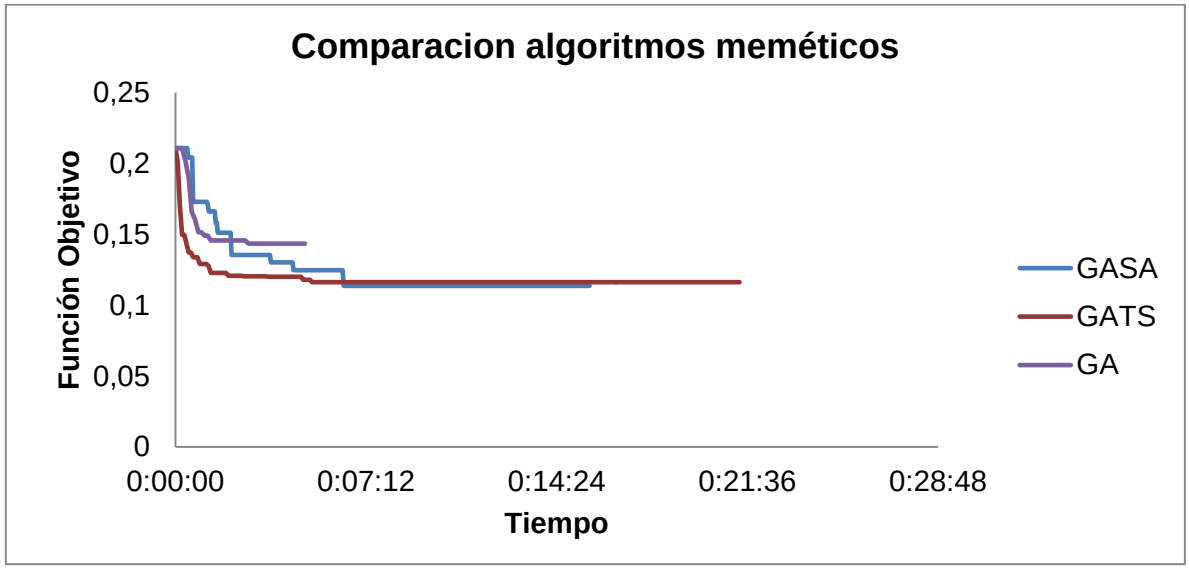


Figura 37. Comparación función objetivo algoritmos meméticos usados para calibración

La Figura 37 ilustra la trayectoria de la función objetivo para las 3 diferentes instancias de algoritmo memético aquí planteadas, dado que el GA no posee un optimizador local se

plantea como causa de su poca mejoría de la función objetivo. Por el contrario nuevamente GASA y GATS alcanzan una mejora significativa en el mismo número de iteraciones. Respecto al comportamiento del SPSA ilustrado en la Figura 38 es notoria la mejora a través del tiempo pero su función objetivo no alcanza un valor tan bajo como el de GASA o GATS.

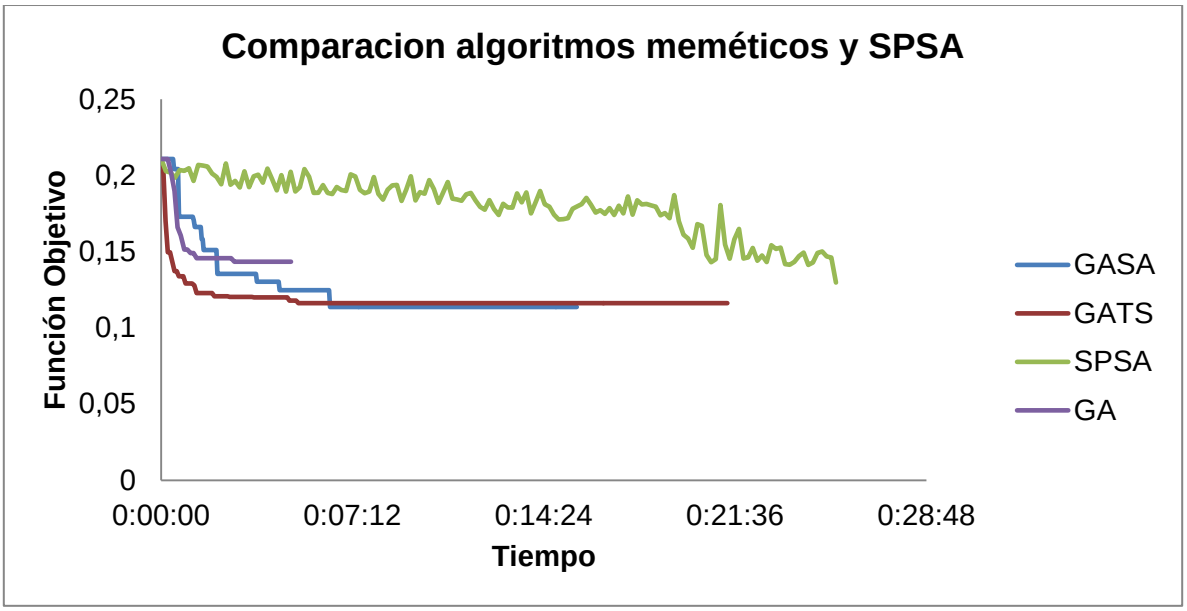


Figura 38. Comparación función objetivo de algoritmos meméticos y SPSA para calibración

La Tabla 2 evidencia como nuevamente el algoritmo que toma un menor tiempo de ejecución es GA, sin embargo es el GASA quien logra el mejor valor de función objetivo en un menor tiempo, sin embargo la diferencia con GATS no es alta y el tiempo de ejecución también es similar. Adicionalmente se puede determinar cómo GASA y GATS superan en un alto grado al SPSA desde el punto de vista de la función objetivo, del tiempo de ejecución y de la estadística GEH.

Tabla 2. Comparativa algoritmos de calibración, modelo Reno

	GA	GASA	GATS	SPSA
NRMS Mínimo	0,1433679	0,113644539	0,11625052	0,12973102
Iteraciones	43	28	28	150
Tiempo Total	00:04:54	00:15:56	00:21:14	00:25:49
GEH	77,7%	93,3%	93,3%	88,8%

6.3 MODELO COMPLEJIDAD ALTA: I-75 NETWORK

El modelo posee 703 enlaces de los cuales se poseen datos de campo para 346; el individuo tiene una longitud de 1121 para este modelo.

6.3.1 Resultados con SPSA

La Figura 39 ilustra el comportamiento de la función objetivo a través dl tiempo. La función objetivo alcanza una mejora desde 0.2971477181 hasta 0,266579982. La mejora no es significativa en comparación con los anteriores modelos debido a la gran cantidad de enlaces de este modelo y el tamaño del individuo. El algoritmo se detiene después del máximo número de iteraciones de 150.

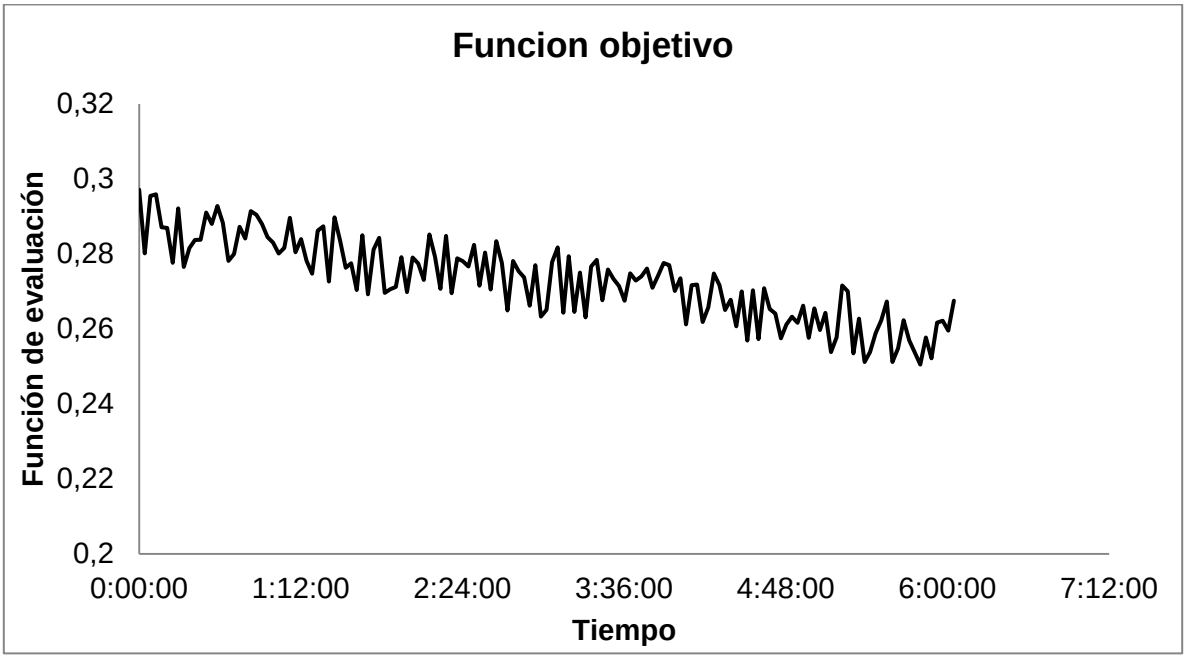


Figura 39. Función objetivo, SPSA, I75

La estadística GEH antes de realizar la calibración es menor a 5 para el 48,2% de los enlaces, después de realizar la calibración es menor a 5 para el 54.1% de los enlaces lo cual evidencia una mejora pero no lo suficiente para considerar el modelo calibrado en base a las directrices dadas por la FHWA.

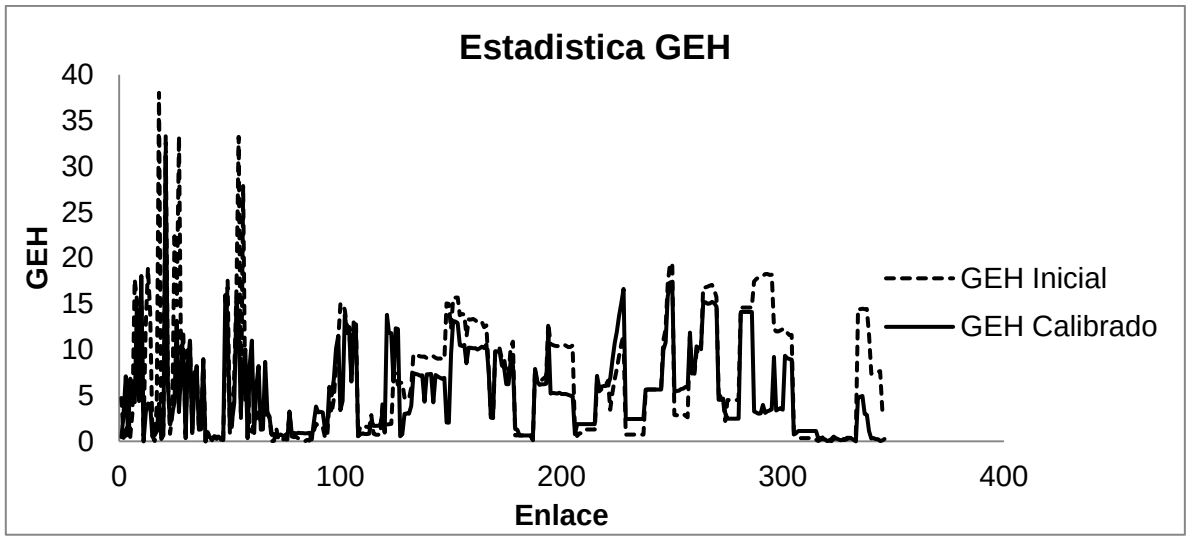


Figura 40. Estadística GEH, SPSA, I75

6.3.2 Resultados con instancia de algoritmo genético sin optimizador local

La función objetivo ilustrada en la Figura 41 evidencia una mejora desde 0.2971477181 hasta 0,266579982, y debido a que no se cumple el criterio de parada dado por la estadística GEH, el algoritmo se detiene dado el criterio del máximo número de paradas sin obtener una mejora. La Figura 44 ilustra como la estadística GEH es mayor a 5 para una gran cantidad de valores, para ser más exactos el GEH es menor a 5 solo para el 52,3% lo cual no es suficiente para considerar el modelo calibrado en base a las directrices de la FHWA.

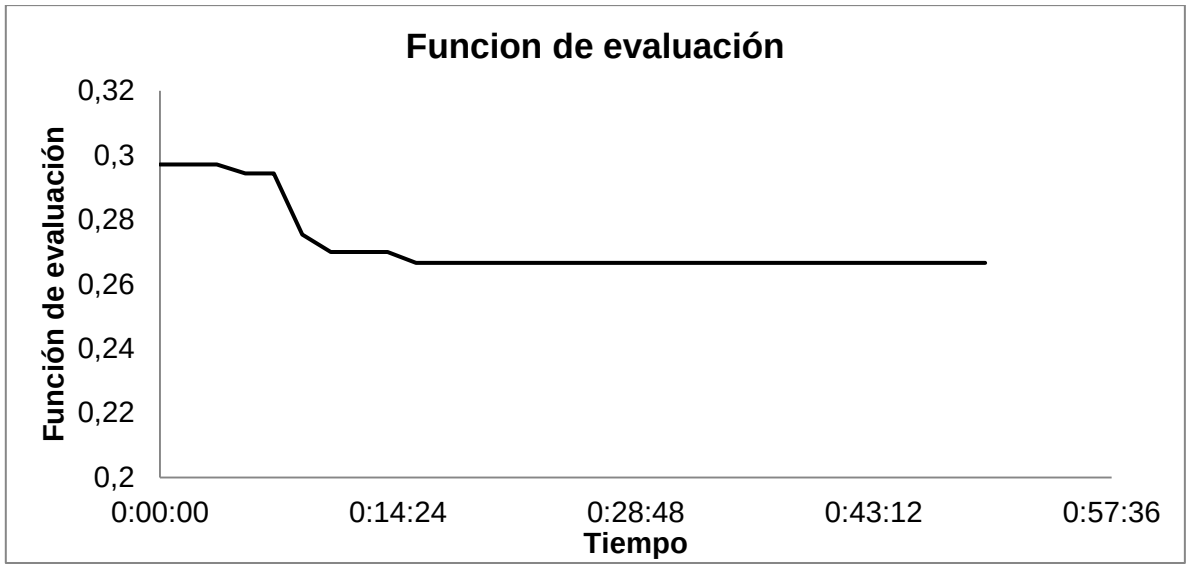


Figura 41. Función objetivo, instancia con algoritmo genético sin optimizador local, I75

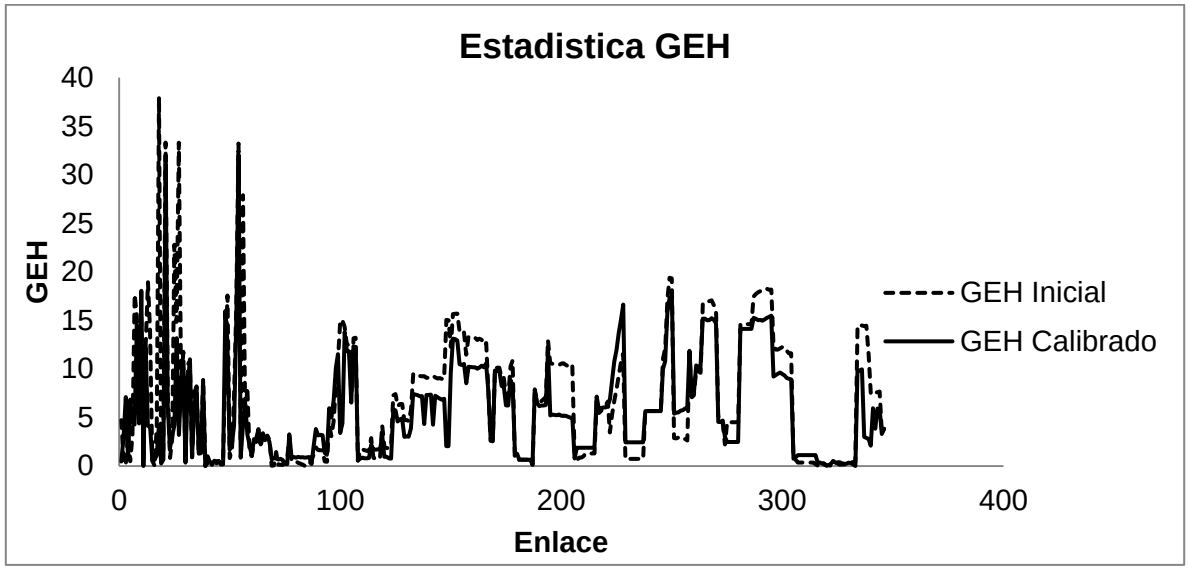


Figura 42. Estadística GEH, instancia con algoritmo genético sin optimizador local, I75

6.3.3 Resultados con instancia de algoritmo genético más recocido simulado

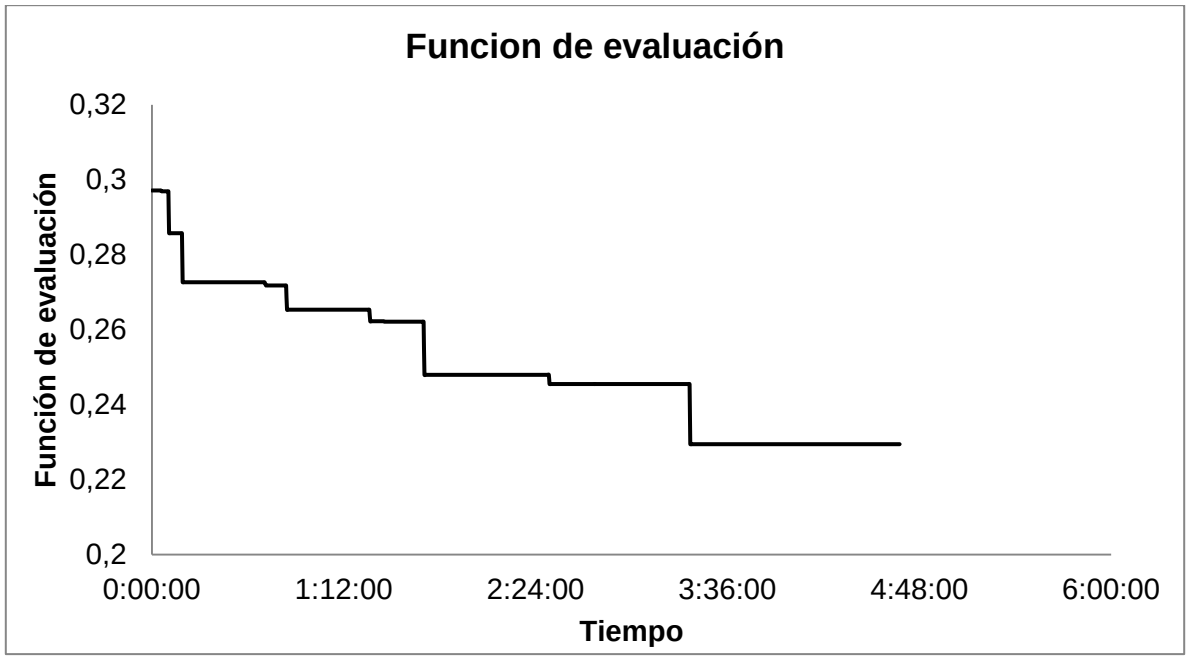


Figura 43. Función objetivo, instancia con algoritmo genético más recocido simulado, I75

La Figura 43 ilustra la trayectoria de la función objetivo a través del tiempo, la cual alcanza una mejora desde 0.2971477181 hasta 0,229458901.

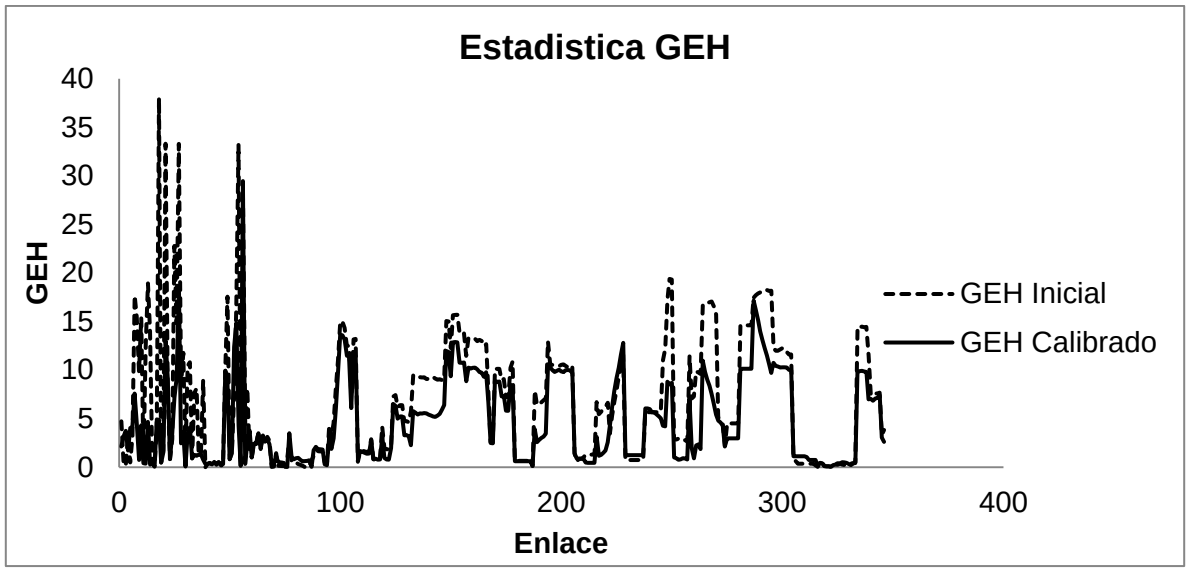


Figura 44. Estadística GEH, instancia con algoritmo genético más recocido simulado, I75

6.3.4 Resultados con instancia de algoritmo genético más búsqueda tabú

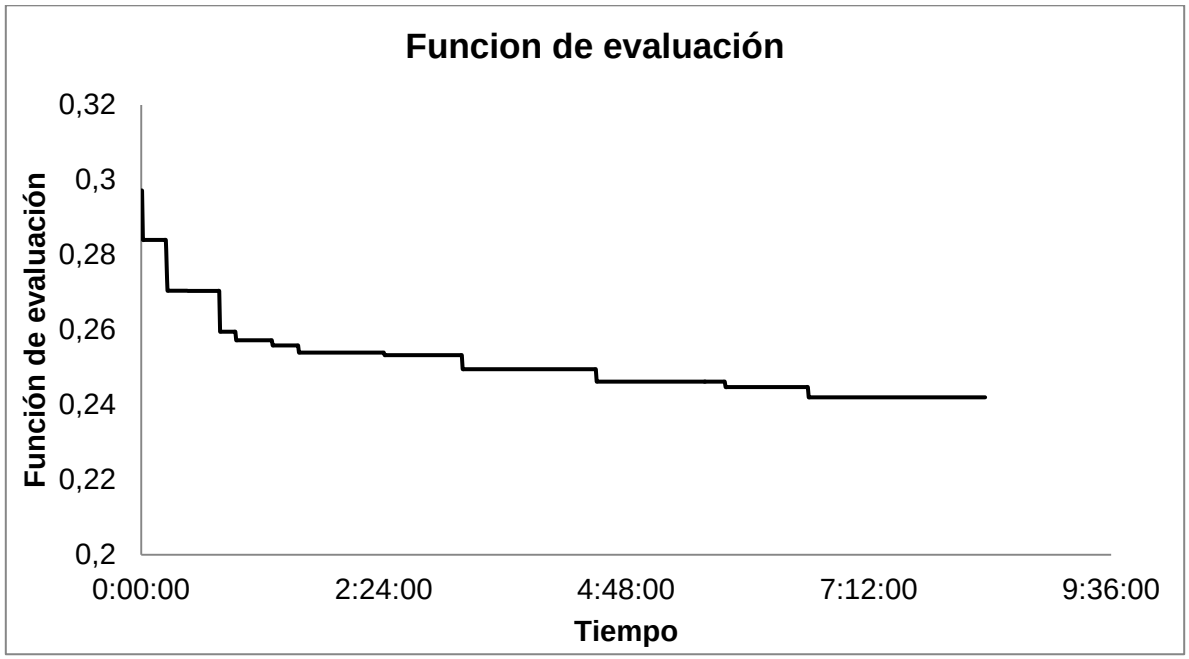


Figura 45. Función objetivo, algoritmo genético más búsqueda tabú, I75

La Figura 46 ilustra la trayectoria de la función objetivo, la cual alcanza una mejora desde 0.2971477181 hasta 0,229458901, esta es la función objetivo con mejor comportamiento para el presente modelo, aunque la estadística GEH evidencia que no se alcanza el mínimo porcentaje requerido.

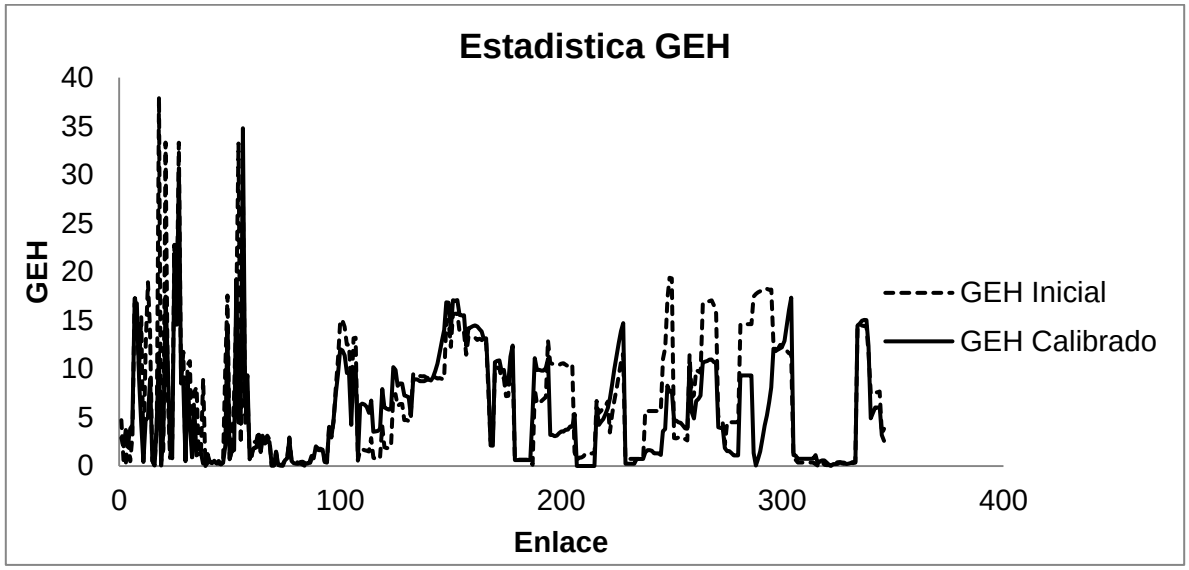


Figura 46. Estadística GEH, instancia de algoritmo genético más búsqueda tabú, I75

La Figura 46 ilustra como a pesar de que existe una mejora en la mayoría de los enlaces para la estadística GEH esta no es suficiente para que se considere el modelo calibrado en base a los lineamientos de la FHWA, después de la calibración el GEH es menor a 5 para el 59,2% de los enlaces que como se ha mencionado antes no es suficiente.

6.3.5 Comparación de algoritmos

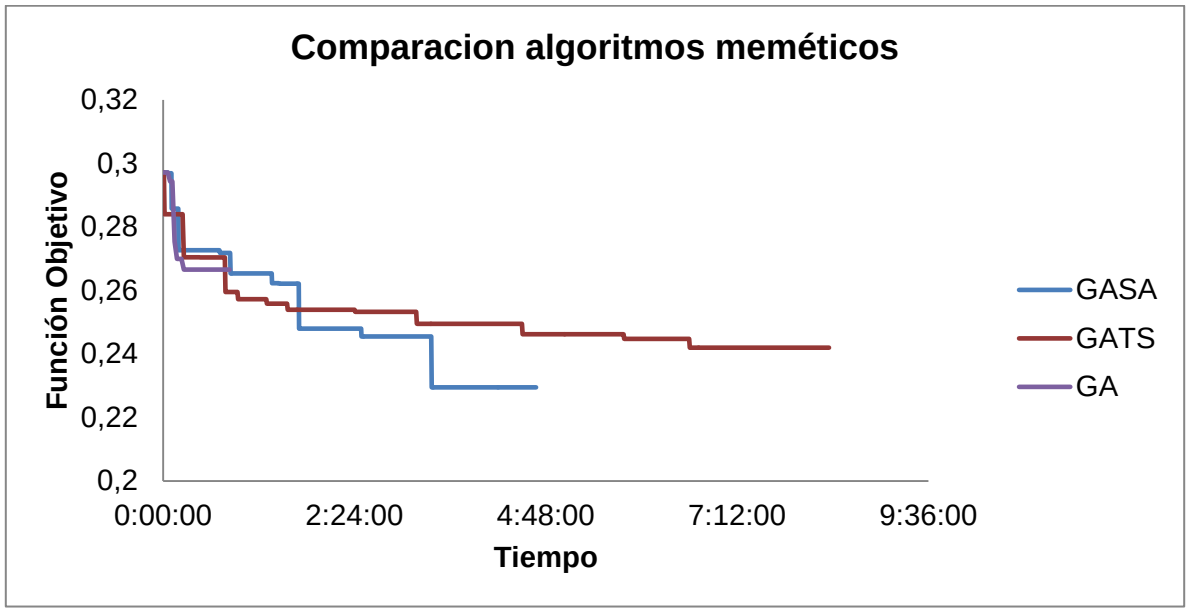


Figura 47. Comparación función objetivo algoritmos meméticos usados para calibración

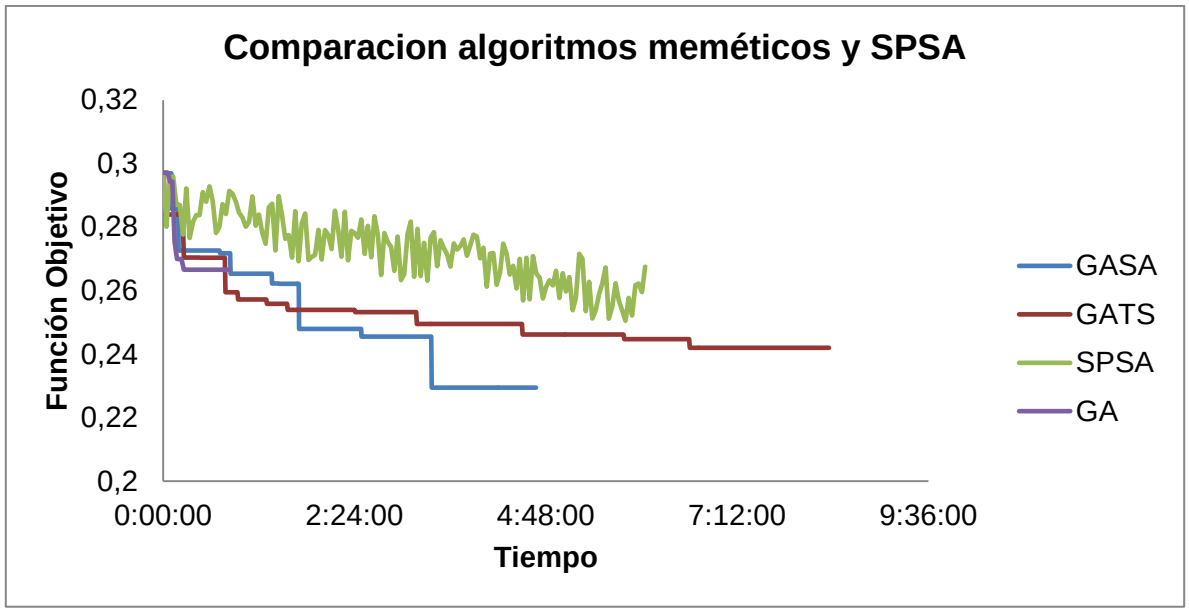


Figura 48. Comparación función objetivo de algoritmos meméticos y SPSA para calibración

La Figura 47 ilustra como de las 3 instancias del algoritmo memético usadas la que tiene un mejor comportamiento para la función objetivo es GASA aunque con una mínima diferencia. La Figura 48 muestra como el algoritmo SPSA logra una mejora notable pero no mayor a GASA y GATS, y de la Tabla 3 se puede deducir como GA toma menos tiempo para realizar la calibración aunque su resultado en función objetivo es el peor de todos, el algoritmo con mejor función objetivo es el GASA y su mejora la alcanza en un tiempo menor que GATS y SPSA. También se puede notar como para este modelo de gran tamaño ninguno de los algoritmos alcanza el criterio de calibración en base a los lineamientos de la FHWA pero si logran una mejora aceptable los modelos.

Tabla 3. Comparativa algoritmos de calibración, modelo I-75

	GA	GASA	GATS	SPSA
NRMS Mínimo	0,266579982	0,229458901	0,241996974	0,25053
Iteraciones	30	46	50	150
Tiempo Total	00:50:16	04:41:23	08:12:36	06:09:20
GEH	52,3%	59,2%	55,4%	54,1%

6.3.6 Discusión de resultados

La tabla 4 muestra los datos de tiempo y precisión obtenidos en la ejecución de cada experimento, dicha tabla se construyó con el fin de valorar la presente discusión de resultados.

La medida de precisión fue obtenida a partir del mejor NRMS alcanzado por cada algoritmo vs el NRMS inicial (el NRMS determina la cercanía entre los datos de campo y datos de la salida de la simulación).

Tabla 4. Resumen de datos de tiempo y precisión de cada algoritmo.

Experimento 1				
	GA	**GASA**	**GATS**	**SPSA**
Tiempo Total	00:01:01	00:04:22	00:09:53	00:12:19
Precisión	68,96%	93,1%	93,1%	82,75%
Experimento 2				
Tiempo Total	00:04:54	00:15:56	00:21:14	00:25:49
Precisión	33,33%	47,61%	47,61%	42,85%
Experimento 3				
Tiempo Total	00:50:16	04:41:23	08:12:36	06:09:20
Precisión	13,33%	26,66%	20%	16,66%

Para todos los experimentos GASA demostró una superioridad en cuanto a equilibrio entre tiempo de calibración y precisión del resultado obtenido, esto debido a que el tiempo de calibración permaneció por debajo del promedio en todos los experimentos, además la precisión obtenida con este algoritmo fue mayor en todos los casos.

GATS demostró la misma precisión que GASA durante los dos primeros experimentos, sin embargo le tomo más tiempo (en el experimento 1 GATS tardo 9 minutos y 53 segundos, mientras que GASA tardo 4 minutos y 22 segundos, en el experimento 2 GATS tardo 21 minutos y 14 segundos mientras que GASA tardo 15 minutos y 56 segundos).

Se puede evidenciar como en las tablas de resultados no existe una relación directa entre el número de iteraciones y el tiempo de calibración, esto es debido a que algoritmos como GASA y GATS tienen un incremento en el tiempo dado por el optimizador local lo que causa que se note que GA siempre tiene un menor tiempo pero un mayor número de iteraciones.

Los algoritmos GASA y GATS logran para los primeros 2 experimentos de tamaño pequeño y mediano, alcanzar el criterio de calibración dado por la FHWA, pero ninguno de los algoritmos logra alcanzar tal criterio para el ultimo experimento de tamaño grande.
Es notorio por las figuras relacionadas con la función objetivo como los 4 algoritmos usados para los experimentos tienen la capacidad de calibrar modelos de micro

simulación de flujo de tráfico vehicular CORSIM, sin embargo las instancias de algoritmos meméticos con optimizador local evidencian mejores resultados.

De manera aproximada se puede establecer una relación lineal entre el tamaño del modelo (basado en cantidad de enlaces y longitud del individuo) y la cantidad de tiempo necesaria para calibrarlo.

Es notorio como el algoritmo GA logra realizar el proceso de calibración con la menor precisión, pero en el menor tiempo (para todos los experimentos), por tanto si en algún contexto de aplicación se puede sacrificar precisión con tal de mejorar tiempo de ejecución GA sería una buena elección, pero si se desea mayor precisión en los resultados, GASA o GATS representan una buena elección.

Capítulo 7

7. CONCLUSIONES Y TRABAJO FUTURO

7.1 CONCLUSIONES

El trabajo realizado permitió evidenciar que la instancia definida de un algoritmo memético tiene la capacidad de calibrar modelos de micro simulación de flujo de tráfico vehicular CORSIM, con alta precisión (93,1%) y bajo los lineamientos dados por el FHWA [6] (GEH por debajo de 5 para más del 85% de los nodos del modelo) modelos de baja complejidad, con una precisión considerable (47,61%) y cumpliendo también los lineamientos de la FHWA modelos de complejidad media, y con una precisión aceptable (26,66%) sin cumplir los estándares de la FHWA modelos de alta complejidad.

Los resultados obtenidos permiten concluir que las instancia del algoritmo memético que de manera general tiene un comportamiento más favorable es el algoritmo memético compuesto por un algoritmo genético y uno de recocido simulado y el algoritmo memético compuesto por un algoritmo genético y uno de búsqueda tabú, ya que para los 3 experimentos estos algoritmos presentan el mejor equilibrio entre precisión del resultado y tiempo de calibración (como se puede observar en la Tabla 4 y la discusión de resultados) por tanto se concluye como estos algoritmos poseen las mejores características para calibrar un modelo de micro simulación de flujo de tráfico vehicular CORSIM.

Los algoritmos meméticos GASA y GATS superan en gran medida los resultados de la calibración provistos por el SPSA, no solo por su evidente mejora en la función objetivo (para todos los casos GASA y GATS es más preciso, Tabla 4) sino porque estos algoritmos meméticos no necesitan un modelo pre calibrado para poder llegar a una solución, sino que pueden encontrar una solución en el vasto espacio de búsqueda propio de los problemas de calibración de modelos de micro simulación de flujo de tráfico vehicular CORSIM.

Para otros tipos de problemas de calibración de modelos de micro simulación diferentes al de tráfico vehicular, es posible aplicar la instancia del algoritmo memético aquí usado siempre y cuando el tipo de modelo a calibrar cumpla las condiciones básicas para aplicar un algoritmo memético planteadas en [55], es decir se pueda definir de manera clara y precisa una función objetivo, un criterio de vecindad y un criterio de parada para el algoritmo.

Una incorrecta definición de la configuración del algoritmo genético, de recocido simulado o búsqueda tabú pueden provocar que las calibraciones aquí planteadas, no funcionen, funcionen de manera errónea o tomen un tiempo alto para ejecutarse, dado que estos parámetros afectan de manera drástica el comportamiento del algoritmo [55].

7.2 RECOMENDACIONES Y TRABAJO FUTURO

Se puede considerar una mejora al algoritmo aplicado con técnicas como los meta algoritmos que tienen la capacidad de encontrar una configuración para el algoritmo memético de tal manera que los resultados de la simulación sean más precisos y el tiempo de calibración sea reducido. También se podría considerar un completo análisis de sensibilidad para optimizar la configuración del algoritmo memético propuesto.

Es posible considerar la implementación para procesamiento en paralelo en un conjunto de máquinas, del algoritmo memético usado ya que uno de los inconvenientes es el tiempo que toma evaluar la función objetivo para lo cual se puede aprovechar el multiprocesamiento. Adicionalmente se puede evaluar la factibilidad de en vez de usar una simulación completa para evaluar la función objetivo, plantear una red neuronal que prediga el comportamiento de la salida de la simulación dado el conjunto de parámetros de entrada y así optimizar el algoritmo propuesto.

Se puede considerar el usar algoritmos de optimización local diferentes (GRASP, Vecindarios variables) a los aquí planteados, que tengan buenas características de explotación dados individuos de gran longitud.

Se puede considerar una investigación enfocada en dividir los modelos de simulación de gran tamaño en varios modelos de tamaño pequeño de tal manera que con la instancia del algoritmo propuesto se puedan lograr resultados de mayor precisión.

Capítulo 8

8. BIBLIOGRAFÍA

[1] E. Tarifa, “Teoría de Modelos y Simulación. Introducción a la Simulación,” *Fac. Ing. Univ. Nac. Jujuy. …*, pp. 1–17, 2005.

[2] M. Zhang and J. Ma, “Developing calibration tools for microscopic traffic simulation final report part 1: Overview methods and guidelines on project scoping and data collection,” *… Path Program, Inst. Transp. Stud. …*, no. January, 2008.

[3] Federal Highway Administration, “CORSIM User’s Guide,” no. December. 2006.

[4] Mctrans, “Reference Manual,” 2010.

[5] R. E. Anderson and C. Hicks, “Highlights of Contemporary Microsimulation,” *Soc. Sci. Comput. Rev.*, vol. 29, no. 1, pp. 3–8, May 2010.

[6] B. Sandoval, “CORSIM Modeling Guidelines,” no. September, 2012.

[7] A. Kondyli, I. Soria, A. Duret, and L. Elefteriadou, “Sensitivity analysis of CORSIM with respect to the process of freeway flow breakdown at bottleneck locations,” *Simul. Model. Pract. Theory*, vol. 22, pp. 197–206, Mar. 2012.

[8] Y. Hollander and R. Liu, “The principles of calibrating traffic microsimulation models,” *Transportation (Amst).*, vol. 35, no. 3, pp. 347–362, Jan. 2008.

[9] J. Yuan, S. H. Ng, and K. L. Tsui, “Calibration of Stochasitc Computer Models Using Stochastic Approximation Methods,” *Autom. Sci. Eng. IEEE Trans.*, vol. 10, no. 1, pp. 171–186, 2013.

[10] J. Ma, H. Dong, and H. M. Zhang, “Calibration of Micro Simulation with Heuristic Optimization Methods,” vol. 45, no. 1, pp. 1–25, 2012.

[11] R. Balakrishna, C. Antoniou, M. Ben-Akiva, H. Koutsopoulos, and Y. Wen, “Calibration of Microscopic Traffic Simulation Models: Methods and Application,” *Transp. Res. Rec.*, vol. 1999, no. 1, pp. 198–207, Jan. 2007.

[12] J. Hourdakis, P. G. Michalopoulos, and J. Kottommannil, “A practical procedure for calibrating microscopic traffic simulation models,” *Transp. Res. Rec.*, vol. 1852, no. January, pp. 1–36, 2003.

[13] F. N. Moreno, “Optimización aplicada a la calibración y validación de modelos de elementos finitos de puentes,” vol. 17, no. 1, pp. 43–59, 2007.

[14] C. Cotta, “Una Visión General de los Algoritmos Meméticos,” 2007.

[15] A. Paz, V. Molano, and C. Gaviria, “Calibration of CORSIM models considering all model parameters simultaneously,” *2012 15th Int. IEEE Conf. Intell. Transp. Syst.*, no. 1, pp. 1417–1422, Sep. 2012.

[16] R. Omrani and L. Kattan, “Simultaneous Calibration of Microscopic Traffic Simulation Model and Estimation of Origin / Destination (OD) Flows based on Genetic Algorithms in a High-Performance Computer * I N ’ iN exs _ XO) 2,” *16th Int. IEEE Annu. Conf. Intell. Transp. Syst.*, pp. 2316–2321, 2013.

[17] B. Park and J. Won, “Microscopic simulation model calibration and validation handbook,” 2006.

[18] L. Cunha, E. B. Jr, and R. Setti, “Genetic Algorithm for the Calibration of Vehicle Performance Models,” *Prog. Artif. Intell.*, vol. 5816, pp. 3–14, 2009.

[19] P. Merz and B. Freisleben, “A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem,” *… , 1999. CEC 99. Proc. …*, no. Fb 12, 1999.

[20] K.-O. Kim and L. Rilett, “Simplex-Based Calibration of Traffic Microsimulation Models with Intelligent Transportation Systems Data,” *Transp. Res. Rec.*, vol. 1855, no. 1, pp. 80–89, Jan. 2003.

[21] S. Mittal and L. Wang, “Design Exploration and Implementation of Simplex Algorithm over Reconfigurable Computing Platforms.”

[22] B. Park and A. Kamarajugadda, “Development and evaluation of a stochastic traffic signal optimization method,” *Int. J. Sustain. …*, 2007.

[23] B. “Brian” Park, I. Yun, and K. Ahn, “Stochastic Optimization for Sustainable Traffic Signal Control,” *Int. J. Sustain. Transp.*, vol. 3, no. 4, pp. 263–284, Jun. 2009.

[24] P. Garg, “A Comparison between Memetic algorithm and Genetic algorithm for the cryptanalysis of Simplified Data Encryption Standard algorithm P10,” *Int. J. Netw. Secur. Its Appl.*, vol. 1, no. 1, pp. 34–42, 2009.

[25] D. Fouskakis and D. Draper, “Stochastic Optimization: a Review,” pp. 315–349, 2002.

[26] G. Lin, W. Zhang, F. Yang, H. Pang, and D. Wang, “An Initial Value Calibration Method for the Wheel Force Transducer Based on Memetic Optimization Framework,” *Math. Probl. Eng.*, vol. 2013, pp. 1–10, 2013.

[27] M. Eusuff, K. Lansey, and F. Pasha, “Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization,” *Eng. Optim.*, vol. 38, no. 2, pp. 129–154, Mar. 2006.

[28] K. Knodler, J. Poland, P. Merz, and A. Zell, “Using Memetic Algorithms for Optimal Calibration of Automotive Internal Combustion Engines,” *Recent Adv. Memetic Algorithms*, vol. 166, pp. 87–104, 2005.

[29] X. Yang, L. Wu, J. Li, and K. Chen, “A minimal kinematic model for serial robot calibration using POE formula,” *Robot. Comput. Integr. Manuf.*, vol. 30, no. 3, pp. 326–334, Jun. 2014.

[30] S. Guillas, N. Glover, and L. Malki-Epshtein, “Bayesian calibration of the constants of the *Comput. Methods Appl. Mech. Eng.*, vol. 279, pp. 536–553, Sep. 2014.

[31] J. Sanyal, J. New, R. E. Edwards, and L. Parker, “Calibrating building energy models using supercomputer trained machine learning agents,” no. March, pp. 2122–2133, 2014.

[32] M. Garcia, I. Ramirez, M. Verlaan, and J. Castillo, “Application of a three-dimensional hydrodynamic model for San Quintin Bay, B.C., Mexico. Validation and calibration using OpenDA,” *J. Comput. Appl. Math.*, vol. 273, pp. 428–437, Jan. 2015.

[33] M. Eshagh and S. Ebadi, “A strategy to calibrate errors of Earth gravity models,” *J. Appl. Geophys.*, vol. 103, pp. 215–220, Apr. 2014.

[34] L. G. Quiroz, Y. Maruyama, and C. Zavala, “Cyclic behavior of Peruvian confined masonry walls and calibration of numerical model using genetic algorithms,” *Eng. Struct.*, vol. 75, pp. 561–576, Sep. 2014.

[35] H. Li, L. Wang, J. Qiu, C. Li, M. Gao, and C. Gao, “Calibration of DNDC model for nitrate leaching from an intensively cultivated region of Northern China,” *Geoderma*, vol. 223–225, pp. 108–118, Jul. 2014.

[36] B. Ciuffo and C. Lima Azevedo, “A Sensitivity-Analysis-Based Approach for the Calibration of Traffic Simulation Models,” *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 3, pp. 1298–1309, Jun. 2014.

[37] J. Zhang, N. Hounsell, and B. Shrestha, “Calibration of bus parameters in microsimulation traffic modelling,” *Transp. Plan. …*, no. September 2014, pp. 37–41, 2012.

[38] A. Kostikj and M. Kjosevski, “Development and calibration of a single lane urban traffic simulator *,” pp. 494–500, 2013.

[39] P. Korcek, L. Sekanina, and O. Fucik, “Evolutionary approach to calibration of cellular automaton based traffic simulation models,” *2012 15th Int. IEEE Conf. Intell. Transp. Syst.*, pp. 122–129, Sep. 2012.

[40] X. Ma, Z. Huang, and H. Koutsopoulos, “Integrated Traffic and Emission Simulation: a Model Calibration Approach Using Aggregate Information,” *Environ. Model. Assess.*, vol. 19, no. 4, pp. 271–282, Jan. 2014.

[41] I. Ištoka Otković, T. Tollazzi, and M. Šraml, “Calibration of microsimulation traffic model using neural network approach,” *Expert Syst. Appl.*, vol. 40, no. 15, pp. 5965–5974, Nov. 2013.

[42] Mctrans, “Tsis Users Guide,” 2010.

[43] K. Eckhardt, N. Fohrer, and H.-G. Frede, “Automatic model calibration,” *Hydrol. Process.*, vol. 19, no. 3, pp. 651–658, Feb. 2005.

[44] M. Zhang and J. Ma, “Developing Calibration Tools for Microscopic Traffic Simulation,” no. January, 2008.

[45] G. Schultz and L. Rilett, “Analysis of Distribution and Calibration of Car-Following Sensitivity Parameters in Microscopic Traffic Simulation Models,” *Transp. Res. Rec.*, vol. 1876, no. 1, pp. 41–51, Jan. 2004.

[46] A. Lisandro, “ALGORITMO DE COLONIA DE HORMIGAS PARA EL ENRUTAMIENTO EN REDES DE SENSORES INALÁMBRICOS,” 2011.

[47] F. S. Hillier, *Integrated Methods for Optimization*. 2012.

[48] F. Neri, C. Cotta, and P. Moscato, *Handbook of Memetic Algorithms*. Springer US, 2012.

[49] M. Mitchell, *An introduction to genetic algorithms*. 1998.

[50] B. Batista and F. Glover, “Introducción a la Búsqueda Tabú,” *mbmelian.webs.ull.es*, vol. 03, pp. 1–36, 2014.

[51] F. Glover and M. Laguna, *Tabu Search*. Boston, MA: Springer US, 1997.

[52] D. Bertsimas and J. Tsitsiklis, “Simulated annealing,” *Stat. Sci.*, vol. 8, pp. 10–15, 1993.

[53] K. S. Pratt, “Design Patterns for Research Methods : Iterative Field Research,” no. 1994, 2009.

[54] J. C. Spall, “An Overview of the Simultaneous Perturbation Method,” vol. 19, no. 4, 1998.

[55] P. Moscato and C. Cotta, “Una Introducción a los Algoritmos Mem éticos,” vol. 19, no. 19, pp. 131–148, 2003.

[56] K. P. Moses, “An Approach to Reduce Root Mean Square Error in Toposheets,” vol. 91, no. 2, pp. 268–274, 2012.

[57] D. E. Goldberg and K. Deb, “A Comparative Analysis of Selection Schemes Used in Genetic Algorithms,” 1991.

[58] R. Rutenbar, “Simmulated Annealing Algorithms An Overview.” 1989.

[59] B. Hajek, “Cooling shedules for optimal annealing,” vol. 13, no. 2, pp. 311–330, 1986.

[60] S. Salhi, “Defining tabu list size and aspiration criterion within tabu search methods,” vol. 29, 2002.