ESCAPE

Emergency System for Calculating Adaptive Pedestrian Evacuations

MANUAL DE USUARIO

Por

Luis Moya

Departamento de Ingeniería, Pontificia Universidad Católica del Perú (PUCP), Perú

Julio Ramírez

Departamento de Ingeniería, Pontificia Universidad Católica del Perú (PUCP), Perú

Erick Mas

Instituto Internacional de Investigación sobre Ciencias de Desastres (IRIDeS), Japón

Shunichi Koshimura

Instituto Internacional de Investigación sobre Ciencias de Desastres (IRIDeS), Japón

Financiado por CONCYTEC-PROCIENCIA (Contrato Número PE501078853-2022)

Septiembre, 2024

INDICE

1. INTRODUCCIÓN

1.1. Presentación del software ESCAPE

El código ESCAPE, utiliza el algoritmo de Sarsa para resolver ecuaciones de Velocidad–Tiempo que ayudará en la demostración de las opciones para la evacuación de peatones en una ciudad, todo ello mediante un algoritmo.

Los principales resultados de la evacuación de cada peatón son los siguientes:

- Tiempo de evacuación
- Posición
- Velocidad

1.2. Características Principales

Las características principales del software son las siguientes:

- El software está codificado con el lenguaje de programación C++, el cual permite que este sea más rápido, debido a que es un código de bajo nivel.
- Permite realizar evacuaciones de ciudades.
- Permite calcular el tiempo total que demora la evacuación de los pobladores de una ciudad.
- Permite ubicar los puntos de destino (evacuación) en una ciudad.

1.3. Contactos

Si se tiene alguna duda durante el uso del software, puede contactarnos enviando un e-mail y tener una asistencia técnica.

Email: ***julio.ramirez@pucp.edu.pe***

2. FUNDAMENTOS

El código SARSA (Estado-acción-recompensa-estado-acción) fue propuesto por Rummery y Niranjan bajo el nombre de "*Modified Connectionist Q-Learning"* (Q-Learning conexionista modificado) (MCQ-L). Siendo Rich Sutton, quien le asigna el nombre de SARSA.

El nombre SARSA se debe a la siguiente cadena:

La función principal para actualizar el valor Q, depende del estado actual del agente "**S_1**", la acción elegida por el agente "**A_1**", la recompensa "**R_2**" que obtiene el agente por elegir esta acción. El estado "**S_2**" en el que entra el agente después de realizar esa acción y, por último, la siguiente acción "**A_2**" que elige el agente en su nuevo estado. Siendo el resultado el acrónimo de la quíntuple (S_t, A_t, R_t+1, S_t+1, A_t+1).

2.1. Algoritmo SARSA

El algoritmo SARSA es el siguiente:

$$Q^{new}(S_t, A_t) \leftarrow (1-\alpha)Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})\right]$$

A un agente SARSA se le conoce como algoritmo de aprendizaje porque interactúa con el entorno y actualiza la política en función de las acciones realizadas. El valor Q de un estado-acción se actualiza mediante un error y se ajusta con un índice de aprendizaje “α”.

Los valores Q representan la posible recompensa recibida en el siguiente paso temporal por realizar la acción a en el estado s, más la recompensa futura descontada recibida de la siguiente observación del estado-acción.

Una ventaja que tiene SARSA es que aprende por sí mismo los valores Q asociados a la adopción de la política que sigue.

2.2. Parámetros SARSA

A continuación, se muestran los parámetros que se utilizan con SARSA:

- **Índice de aprendizaje (α):** Determina hasta qué punto la información recién adquirida anula la antigua. Un factor 0 hace que el agente no aprenda nada, por otro lado, el factor 1 hace que el agente tenga en cuenta la información más reciente.

- **Factor de descuento (gamma)**: Determina la importancia de las recompensas futuras. Un factor de descuento 0 hace que el agente sea "oportunista", o "miope", por ejemplo, ya que sólo tiene en cuenta las recompensas actuales, mientras que un factor cercano a 1 hará que se esfuerce por obtener una recompensa elevada a

largo plazo. Si el factor de descuento es igual o superior a 1, el valor Q puede divergir.

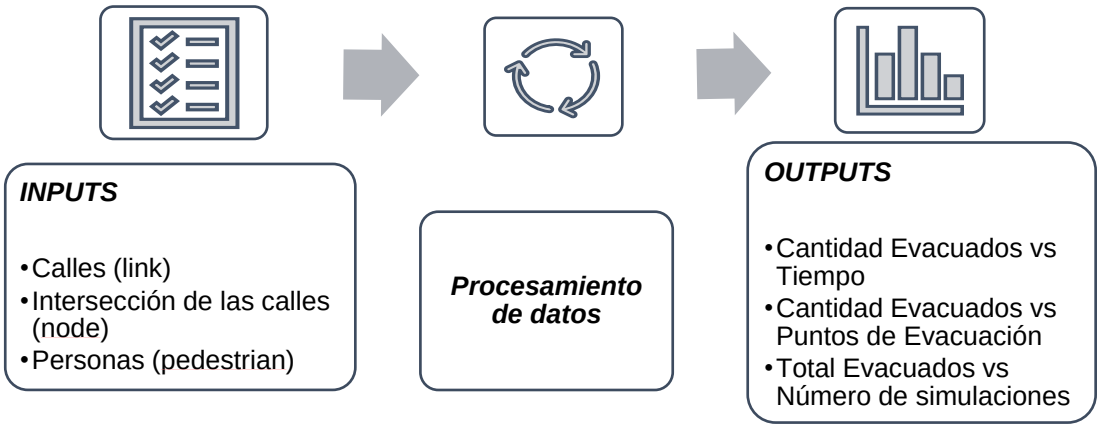
- **Condiciones iniciales ($Q(S_0, A_0)$)**: Dado que SARSA es un algoritmo iterativo, antes de que se produzca la primera actualización, asume una condición inicial. Un valor inicial alto o infinito, también conocido como "condiciones iniciales optimistas", puede fomentar la exploración: no importa qué acción tenga lugar, la regla de actualización hace que tenga valores más altos que la otra alternativa, aumentando su probabilidad de elección. La primera vez que se realiza una acción, la recompensa “r” restablece las condiciones iniciales y se utiliza para fijar el valor de Q. Esto permite un aprendizaje inmediato en caso de recompensas deterministas.

3. ESTRUCTURA Y ALGORITMO DEL PROGRAMA

3.1. Estructura del Programa

El software ESCAPE se ejecuta de acuerdo con la estructura que se muestra en la siguiente figura. Se muestra la información de entrada que necesita el programa, como las calles (link), intersecciones de las calles (node) y las personas (pedestrian), cada una con su identificador. Una vez se introduzca toda la información se realiza el procesamiento de datos que consiste.

Luego de procesar de datos, se tiene como salidas la demostración en tablas y/o figuras de la cantidad de evacuados vs tiempo, cantidad de evacuados vs puntos de evacuación y el total de evacuados vs número de simulación.



3.2. Módulos del Software

3.2.1. Módulo Node (intersecciones)

El módulo Node presenta los siguientes miembros:

- *idNode:* son los identificadores únicos de cada intersección
- *coordenada:* contiene el vector de ubicación de la intersección
- *linkConnectionsPtr:* es un vector de las calles interconectadas
- *stateMatrixsExperimentadosPtr:* es un vector de los estados experimentados

Los miembros *idNote* y coordenada se identifican con el archivo de intersecciones; mientras que, *linkConnectionsPtr* y *stateMatrixsExperimentadosPtr,* son miembros que se obtienen a medida que el software procesa los datos.

Asignación de valores de intersección

Para introducir los valores de las intersecciones se necesita un archivo donde se encuentren estos valores cuyo nombre por default es node.csv, se utiliza una función “constructor" que necesita como parámetros el identificador de

la intersección y el vector ubicación. El archivo de control *controlDict* tiene el comando *nodeFile* para especificar el nombre del archivo que por default es node.csv.

A continuación, se muestra un ejemplo del cambio de nombre del archivo de intersecciones:

```
nodesFile: interseccionesLima.csv
```

3.2.2. Módulo Link (calles)

El módulo Link presenta los siguientes miembros:

- *idLInk:* Identificador único de una calle. Tiene el nombre por default link.csv.
- *node1Ptr:* Una intersección de la calle.
- *node2Ptr:* Otra intersección de la misma calle.
- *length:* Largo de la calle
- *width:* Ancho de la calle
- *orientacionLInk:* Orientación de la calle.
- *anchoSubdivision:* Ancho de la calle.
- *cantidadSubdivisiones:* Cantidad de subdivisiones de las calles, para el cálculo de densidad.
- *densityLevel:* La densidad de la calle. Es calculada con la máxima densidad de las subdivisiones.
- *subdivisiones:* Vector de la clase *subLink*, que es la clase que representa una división.

Asignación de valores de calles

Para introducir los valores de las intersecciones se necesita un archivo donde se encuentren estos valores cuyo nombre por default es link.csv, se utiliza una función “constructor" que necesita como parámetros el ID de la calle, dos puntos de inicio y fin de la calle, la longitud y el ancho de la calle. El archivo de control *controlDict* tiene el comando *linkFile* para especificar el nombre del archivo que por default es link.csv.

A continuación, se muestra un ejemplo del cambio de nombre del archivo de calles:

```
linkFile: callesLima.csv
```

Opciones para la subdivisión de una calle

Existen dos opciones para subdividir una calle: *anchoSubdivisión* o *cantidadSubdivisiones.*

Por default se encuentra activado la opción *anchoSubdivisión*. Esta opción permite dividir la calle de acuerdo a una longitud fija de cada subdivisión y con ello se calcula la cantidad de subdivisiones. En caso la cantidad de subdivisiones no es un entero se procede a redondear y se vuelve a calcular

un nuevo ancho, el cual es muy cercano al anterior, esto se realiza debido a que no se puede tener una cantidad de subdivisiones con valor decimal.

Por otro lado, la segunda opción permite ingresar un número de divisiones a la calle y con ello se calculará un ancho único cada subdivisión.

A continuación, se muestra un ejemplo de la primera opción que aparece por default para la subdivisión de una calle:

```
opcionSubdivision: anchoSubdivision
```

Ancho de subdivisiones

En caso la elección de modo de división de calle haya sido *anchoSubdivision,* la primera opción que se tiene por default, existe el comando *anchoSubdision* que permite colocar el valor de la longitud de las subdivisiones. Los valores pueden ser enteros o decimales, pero no fracciones.

A continuación, se muestra un ejemplo de la primera opción que aparece por default para la subdivisión de una calle:

```
anchoSubdivision: 12.5
```

Cantidad de subdivisiones

Para poder utilizar la segunda opción para subdividir una calle, se debe activar de forma obligatoria el comando *opcionSubdiviones,* como se muestra en el siguiente ejemplo:

```
 opcionSubdiviones:                                    cantidadSubdivisiones
cantidadSubdivisiones: 20
```

3.2.3. Módulo *nodeDestino* (Puntos de evacuación)

El módulo *nodeDestino* presenta los siguientes miembros:

- *maxPersonasEvacuadas*: Es la cantidad de personas que puede evacuar en este sitio. Por default esto está desactivado, por lo tanto, cada punto de evacuación tiene una capacidad infinita; para activar esta opción y limitar la capacidad, se debe modificar el archivo de intersecciones colocando el valor 2 y a continuación la cantidad máxima de evacuados para ese punto.

Impresión de imagen de Evacuados vs Tiempo para diferentes simulaciones

Muestra una línea con la información del total de personas evacuadas en todo el tiempo de evacuación para ciertas simulaciones. Se agrega el número

deseado de simulación con el comando *totalEvacuadosVsSimulacionAt*, mostrando cómo evoluciona el aprendizaje del algoritmo. Esta información se exporta en formato png.

A continuación, se muestra un ejemplo de activación de figura:

```
figureTotalEvacuadosVsTiempo: yes
```

Impresión de tabla de Evacuados vs Tiempo para diferentes simulaciones

Imprime una tabla con los valores de total de evacuados en todo el tiempo de evacuación para ciertas simulaciones. Se agrega el número deseado de simulación con el comando *totalEvacuadosVsSimulacionAt,* mostrando cómo evoluciona el aprendizaje del algoritmo. Esta información se exporta en formato csv.

A continuación, se muestra un ejemplo de activación de tabla:

```
tableTotalEvacuadosVsTiempo: yes
```

Evacuación vs Tiempo para simulaciones específicas

La impresión de la tabla y la figura del total de evacuados vs tiempo utiliza los números de simulaciones 1, 2 y 3 por defecto. Para cambiar estos valores se puede hacer uso del comando *totalEvacuadosVsTiempoAt,* en los cuales se debe asignar valores enteros separados por comas.

A continuación, se muestra un ejemplo de la asignación de simulaciones:

```
evacuadosVsTiempoAt 1, 5, 8;
```

Impresión de imagen del Total de evacuados Vs Simulación

Muestra una gráfica de una línea con la información del total de personas evacuadas a lo largo de las simulaciones. Mostrando cómo evoluciona el aprendizaje del algoritmo. Esta información se exporta en formato png.

A continuación, se muestra un ejemplo de activación de figura:

```
figureTotalEvacuadosVsSimulacion: yes
```

Impresión de tabla del Total evacuados Vs Simulación

Muestra una tabla con los valores de total de evacuados a lo largo de las simulaciones. Mostrando cómo evoluciona el aprendizaje del algoritmo. Esta información se exporta en formato csv.

A continuación, se muestra un ejemplo de activación de tabla:

```
tableTotalEvacuadosVsSimulacion: yes
```

Periodo de simulaciones del Total de evacuación vs Simulación

La impresión de la tabla y la figura del total de evacuados vs Simulación se realiza cada simulación, el cual es un valor por defecto. Para cambiar este valor se puede hacer uso del comando *totalEvacuadosVsTSimulacionPerido*, en los cuales se debe asignar el valor de cada cuanto se desean valores de simulación.

A continuación, se muestra un ejemplo de la asignación del periodo de simulaciones:

```
totalEvacuadosVsSimulacionPeriod 5;
```

3.2.4. Módulo Tiempo

El módulo Tiempo presenta los siguientes miembros:

- *valorTiempo:* Valor del tiempo simulado que aumenta según el *deltaT*.
- *deltaT:* Valor de paso del tiempo. Por default es 1 segundo.
- *startNumberSimulation:* Es el valor de la simulación inicial. Por default es 0.
- *endNumberSimulation:* Es el valor de la última simulación. Este debe ser ingresado en el archivo de *control.l.*
- *graphicPrintouPeriod:* Valor de cada cuanto debería exportar las variables de posición, velocidad y cantidad de personas evacuadas.
- *pedestrianCountPeriod:* Valor de cada cuanto debería calcularse la densidad.

Tiempo de evacuación

Este valor sirve para asignar de manera obligatoria el tiempo de evacuación en segundos. A continuación, se muestra un ejemplo:

```
endTime  800;
```

Colocación del número de simulaciones

Este valor sirve para asignar de manera obligatoria el número de simulaciones. Es decir, cada simulación realiza una evacuación. Solo se aceptan valores enteros y mayor a 0. A continuación, se muestra un ejemplo:

```
endNumberSimulation  7000;
```

Periodo de impresión de variables

Este comando permite imprimir las variables cada cierto tiempo, el cual es asignado en segundos. Por ejemplo, si el comando tiene el valor de 5 y el comando *endTime* 50, entonces se imprimirán las variables en la carpeta data en los siguientes tiempos: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 por default.

A continuación, se muestra un ejemplo:

```
graphicPrintoutPeriod 5;
```

Periodo de cálculo de densidad

Este comando permite realizar el conteo de las personas en cada subdivisión de la calle para luego poder calcular la densidad de cada subdivisión y obtener el máximo valor para asignarlo a la calle con el nivel de densidad. Por ejemplo, si el comando tiene el valor de 6 y el comando *endTime* 50, entonces se calculará la densidad en los tiempos 0, 6, 18, 24, 30, 36, 42 y 48.

A continuación, se muestra un ejemplo:

```
pedestrianCountPeriod 6;
```

4. HERRAMIENTAS

4.1. Lista de Comandos

A continuación, se muestra una lista de los comandos del programa SARSA, los cuales son importantes para modificar parámetros:

4.1.1. nodesFile

- ➔ Tipo: string
- ➔ Valor por defecto: *nodes.csv*

Nombre del archivo de entrada para la lectura de intersecciones. El formato del archivo debe ser csv.

4.1.2. linksFile

- ➔ Tipo: string
- ➔ Valor por defecto: *links.csv*

Nombre del archivo de entrada para la lectura de calles. El formato del archivo debe ser csv.

4.1.3. populationsFile

- ➔ Tipo: string
- ➔ Valor por defecto: *population.csv*

Nombre del archivo de entrada para la lectura de los peatones. El formato del archivo debe ser csv.

4.1.4. sarsaProcesses

- ➔ Tipo: string
- ➔ Valor por defecto: *calibration*

Existen dos procesos para el modelamiento de evacuación. Las opciones disponibles son las siguientes:

- *Calibration:* En este proceso el código necesita entrenar al algoritmo SARSA para que tome sus propias decisiones. El proceso de calibración permite explorar y experimentar decisiones para almacenar los resultados del aprendizaje, este proceso puede repetirse según la cantidad de simulaciones deseadas.

- *Trained:* En este proceso el algoritmo SARSA utiliza lo que aprendió. Solo tiene permite utilizar el aprendizaje anterior y en base a ello utilizarlo para modelar una simulación.

4.1.5. stopAt

- ➔ Tipo: string
- ➔ Valor por defecto: *endTime*

Da el tipo de opciones para parar el tiempo de evacuación.

4.1.6. stopSimulationAt

- ➔ Tipo: string
- ➔ Valor por defecto: *endNumberSimulation*

Da el tipo de opciones para parar terminar la cantidad de simulaciones.

4.1.7. startTime

- ➔ Tipo: int
- ➔ Valor por defecto: 0

Es el tiempo de inicio de la evacuación, este debe ser ingresado en segundos.

4.1.8. endTime

- ➔ Tipo: int
- ➔ Valor por defecto: no tiene

Es para introducir el último tiempo de evacuación en el que se requiere realizar la simulación. El tiempo debe ser ingresado en segundos.

4.1.9. deltaT

- ➔ Tipo: int
- ➔ Valor por defecto: 1

Es el intervalo de tiempo para realizar los cálculos, este debe ser ingresado en segundos.

4.1.10. meanRayleight

- ➔ Tipo: int
- ➔ Valor por defecto: 420

Es el número promedio de la distribución *Rayleight* (único parámetro modificable de la distribución). Este permite calcular el factor de escala para obtener un número según la distribución mencionada. El valor debe ser ingresado en segundos.

4.1.11. exploration

- ➔ Tipo: double
- ➔ Valor por defecto: cercano a 0.2

Es el número de exploración final. Al transcurrir las simulaciones, el ratio de exploración irá cambiando hasta llegar al número ingresado.

4.1.12. endNumberSimulation

- ➔ Tipo: int
- ➔ Valor por defecto: no tiene

Es la cantidad de simulaciones que se realizará, se introduce el último tiempo de simulación. Tener en cuenta que es obligatorio colocar este valor.

4.1.13. graphicPrintout

➔ Tipo: bolean
➔ Valor por defecto: Yes

Sirve para exportar las variables de posición, velocidad, cantidad de evacuados.

Las opciones válidas son las siguientes:

- No
- Yes

Tener en cuenta que las variables deben tener el formato csv.

4.1.14. graphicPrintoutPeriod

➔ Tipo: int
➔ Valor por defecto: 1

Es el intervalo de tiempo que se debe introducir para saber cada cuanto tiempo se exportan las variables. Es el intervalo de exportación. Puede ser mayor que 1. Debe colocarse en segundos.

4.1.15. pedestrianCountPeriod

➔ Tipo: int
➔ Valor por defecto: 1

Es el intervalo de conteo de personas para el cálculo de densidad de una determinada zona. Puede ser mayor que 1. Debe colocarse en segundos.

4.1.16. computationContinued

➔ Tipo: bolean
➔ Valor por defecto: No

En caso se desee leer una base de datos de resultados aprendidos.

Las opciones válidas son las siguientes:

- Yes
- No

4.1.17. previousComputationFile

➔ Tipo: string
➔ Valor por defecto: no tiene

Nombre del archivo de entrada para la lectura de resultados aprendidos en simulaciones anteriores. El formato del archivo debe ser csv.

4.1.18. observationStatePedestrian

➔ Tipo: boolean
➔ Valor por defecto: Yes

Las opciones válidas son las siguientes:

- No
- Yes

Permite contar la cantidad de peatones que experimenta un determinado estado y acción. En caso la opción sea “No”, las futuras simulaciones no tendrán un valor correcto, debido a que no se estará al transcurrir sus simulaciones.

4.1.19. numberLinkDivision

- ➔ Tipo: int
- ➔ Valor por defecto: 10

Es el número de particiones que se le realizan a las calles para obtener un conteo de peatones.

4.1.20. evacuatedCount

- ➔ Tipo: boolean
- ➔ Valor por defecto: Yes

Las opciones válidas son las siguientes:

- Yes
- No

Permite exportar un archivo con la cantidad de personas evacuadas por nodo de evacuación en el tiempo de evacuación del proceso de *“trained”.*

4.1.21. totalEvacuatedCount

- ➔ Tipo: boolean
- ➔ Valor por defecto: Yes

Las opciones válidas son las siguientes:

- Yes
- No

Permite exportar un archivo con la cantidad de personas evacuadas totales en el tiempo de evacuación del proceso de *“trained”.*

4.1.22. StopSimulationAt

- ➔ Tipo: string
- ➔ Valor por defecto: *endNumberSimulation*

Es la forma de detener las simulaciones en el modelo. Solo una opción es válida.

4.1.23. pythonVersion

- ➔ Tipo: boolean
- ➔ Valor por defecto: No

Las opciones válidas son las siguientes:

- Yes
- No

Permite utilizar datos de aprendizajes obtenidos de la versión de Python.

5. ENTRADAS Y SALIDAS

5.1. Observaciones preliminares

El software ESCAPE utiliza un conjunto de archivos como entradas o salidas. Algunos de estos archivos son opcionales.

Los archivos de entrada son los siguientes:

- Intersecciones
- Calles
- Personas
- Control

Los archivos de salida son los siguientes:

- Cantidad de Evacuados vs Tiempo
- Cantidad de Evacuados vs Puntos de Evacuación
- Total de Evacuados vs Número de Simulación

5.2. Los archivos

5.2.1. Archivo de intersecciones

El usuario puede cambiar el nombre del archivo utilizando el comando *nodesFile*. El archivo contiene lo siguiente:

- Id de la intersección, número único y no puede ser igual a otra intersección.
- Coordenada x
- Coordenada y

```
# Ejemplo del archivo nodes.csv

# id x y

0,0,0,0,1

1,20,0,0,1

2,40,0,0,1

3,60,0,0,1

4,80,0,0,1
```

El orden no es importante. Las líneas que empiezan con # son tomados como comentarios. La cantidad de puntos es libre y no necesariamente deben ser las mismas al ejemplo.

5.2.2. Archivo de calles

El usuario puede cambiar el nombre del archivo utilizando el comando *linksFile*. El archivo contiene lo siguiente:

- Id de la calle, número único y no puede ser igual a otra calle.
- Id de la intersección de inicio.
- Id de la intersección de salida.

Ejemplo del archivo *nodes.csv id,node1,node2,widht,*

```
0,0,1,20,3

1,1,2,20,3

2,2,3,20,3

3,3,4,20,3

4,4,5,20,3
```

El orden no es importante. Las líneas que empiezan con # son tomados como comentarios. La cantidad de calles es libre y no necesariamente deben ser las mismas al ejemplo.

5.2.3. Archivo de personas

El usuario puede cambiar el nombre del archivo utilizando el comando *populationsFile*.

El archivo contiene lo siguiente:

- Edad de la persona
- Género de la persona, si hombre es 1.
- Categoría HHType
- Categoría HHID
- Intersección de inicio de la persona.

```
# Ejemplo del archivo population.csv

# age,gender,HHType,HHID,closeNode

18,1,0,0,26

18,1,0,0,22

18,1,0,0,20

18,1,0,0,6
```

El orden no es importante. Las líneas que empiezan con # son tomados como comentarios. La cantidad de personas es libre y no necesariamente deben ser las mismas al ejemplo.

5.2.4. Archivo de control

El archivo representa un panel de control del código ESCAPE. Contiene un número de comandos donde los valores pueden ser asignados o, caso contrario, se usarán los valores por defecto (“Default”). Todos los comandos están definidos en manual de referencias.

```
nodesFile nodes.csv;

linksFile links.csv;

populationsFile population.csv;

stopAt endTime;

endTime 800;

deltaT 1;

graphicPrintoutPeriod 1;

listingPrintoutPeriod 1;

computationContinued no;

previousComputationFile sim_000006000.csv;

# stopSimulationAt

endNumberSimulation;

# endNumberSimulation 6003;

stopSimulationAt

addNumberSimulation;

addNumberSimulation 3;
```

Las líneas que empiezan con # son tomados como comentarios.

5.2.5. El archivo de salida: Cantidad de Evacuados vs Tiempo

En formato csv se exporta la información de la cantidad de evacuados en el tiempo de evacuación, solo para el proceso de *trained*. A continuación, se muestra un ejemplo de este formato.

```
1,0
2,0
3,0
4,0
5,0
6,0
7,0
8,0
9,0
10,0
11,0
12,0
13,0
14,0
15,0
16,0
17,0
18,0
19,0
20,0
21,0
22,0
23,0
24,0
25,0
26,0
27,0
28,0
29,0
30,0
31,0
32,0
33,0
34,0
35,0
36,0
37,0
38,0
39,0
40,0
41,0
42,0
43,0
44,0
45,0
46,0
47,0
48,0
```

La primera columna son datos de tiempo en la evacuación y la segunda columna es la cantidad total de evacuados.

Se puede desactivar con el siguiente comando:

```
totalEvacuatedCount no;
```

5.2.6. El archivo de salida: cantidad de Evacuados vs Puntos de Evacuación

En formato csv se exporta la información de la cantidad de evacuados en cada punto de evacuación en el tiempo, solo para el proceso de *trained*. A continuación, se muestra un ejemplo de este formato.

```
id,35
1,0
2,0
3,0
4,0
5,0
6,0
7,0
8,0
9,0
10,0
11,0
12,0
13,0
14,0
15,0
16,0
17,0
18,0
19,0
20,0
21,0
22,0
23,0
24,0
25,0
26,0
27,0
28,0
29,0
30,0
31,0
32,0
33,0
34,0
35,0
36,0
37,0
38,0
39,0
40,0
41,0
42,0
43,0
44,0
45,0
46,0
47,0
```

La primera columna muestra el tiempo de evacuación mientras que en la segunda columna se muestra:

- En la primera fila: El número de identificación del nodo
- En las siguientes filas: La cantidad de evacuados en ese nodo

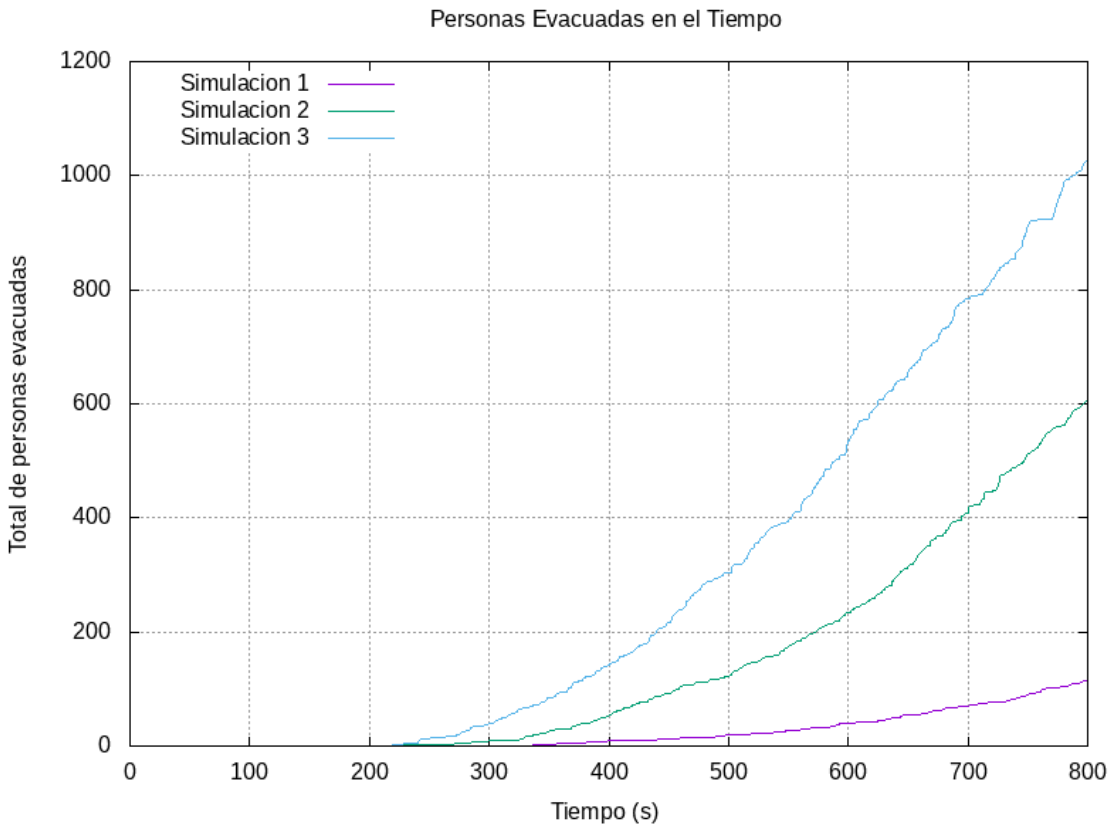
Se puede desactivar con el siguiente comando:

```
evacuatedCount no;
```

5.2.7. El archivo de salida: Total de personas evacuadas vs Tiempo para diferentes simulaciones

Muestra la cantidad de personas evacuadas en el tiempo de una simulación específica. Se creará el gráfico de "*Total de Personas Evacuadas vs tiempo"* para diferentes simulaciones, con el fin de observar la calibración del algoritmo, este se crea con el proceso de *calibration*.

Por default se encuentra activado para que se muestre la figura.



En caso se desee desactivar, se realiza con el siguiente comando:

```
figureTotalEvacuadosXTiempo no;
```

6. SIMULACIÓN DE LA EVACUACIÓN

6.1. Prescripción de condiciones iniciales

Existen dos condiciones iniciales para iniciar la simulación de la evacuación:

- Las posiciones donde están ubicadas las personas, esta condición no es modificable mediantes en el archivo de nodos
- El tiempo que van a salir para evacuar, es una condición inicial que se basa en la distribución de Rayliegh en donde el usuario puede cambiar el parámetro.

6.2. Inicio del Programa

El programa se invoca escribiendo el comando "*Sarsa"* en la terminal. El comando ejecuta un archivo binario guardado en la carpeta "bin", que es un binario de todos los scripts de la carpeta "*src"*.

La sintaxis del comando es la siguiente:

sarsa

6.3. Ejemplos

ESCAPE incluye tres casos tutoriales para mostrar las funcionalidades del código. Estos se encuentran disponibles en la carpeta "*tutorials"*.

6.3.1. Caso grilla

Este ejemplo mostrará la ruta de evacuación más óptima cuando las personas salen de un mismo punto inicial hasta su punto de evacuación.

La configuración de la ciudad presenta calles con un patrón característico que son uniones de 4 calles que forman un rectángulo, con un total de 25 bloques con la misma forma, como se muestra en la siguiente imagen.

Asimismo, una calle tiene una longitud de 20 metros y un ancho 3 metros. Esta configuración presenta un único punto de evacuación ubicado en la posición (100, 100).

El usuario debe colocar cierta información en el *controlDict*:

- Tiempo final de evacuación con el comando *endTime*.
- El número final de simulaciones con el comando *endNumberSimulation*.
- Tipo de proceso con el comando *process*.
- El valor medio para la distribución de Rayleigh.
- El nombre del archivo de población.

Para el presente caso se muestra en la siguiente imagen el archivo mencionado anteriormente, así como los valores de los comandos obligatorios.

```
nodesFile        nodes.csv;

linksFile        links.csv;

populationsFile  populationAtPoint.csv;

sarsaProcesses calibration;

# sarsaProcesses trained;

stopAt          endTime;

endTime         800;

deltaT          1;

meanRayleigh   420;

computationContinued    no;

previousComputationFile calibrated.csv;

stopSimulationAt          endNumberSimulation;

endNumberSimulation 500;

observationStatePedestrian yes;

numberLinkDivision 10;

raphicPrintoutPeriod 1;

listingPrintoutPeriod 1;

evacuatedCount yes;

totalEvacuatedCount yes;

```

Existen comandos que son necesarios especificar su valor debido a que estará utilizando su valor por defecto, el cual está listado en la sección de comandos.

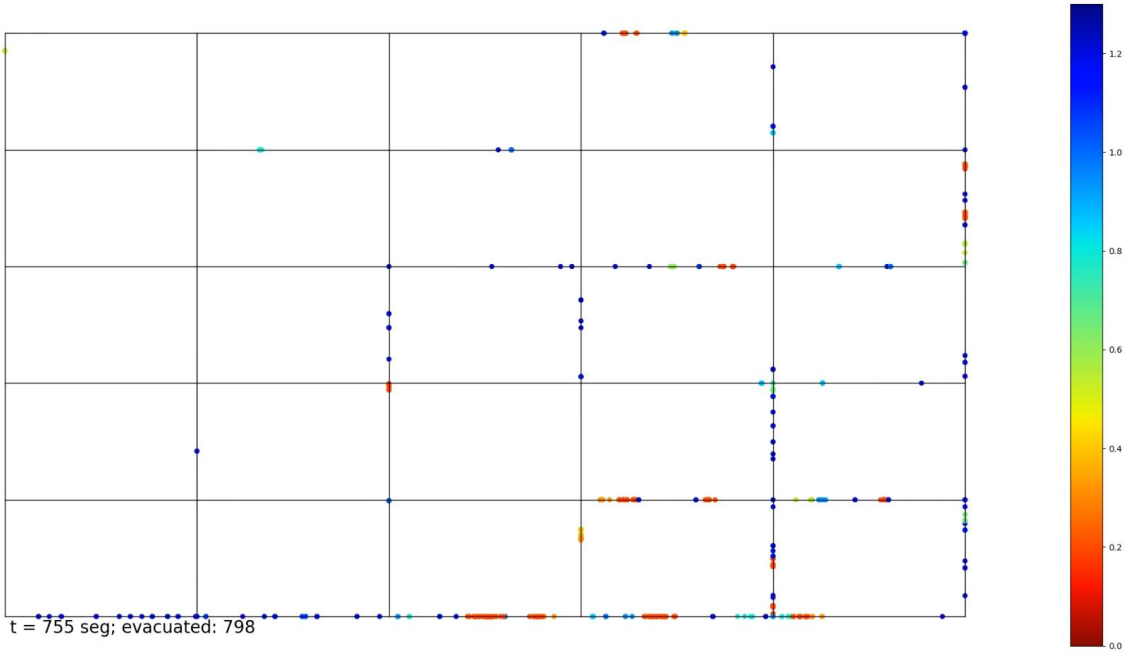
Una vez definido el *controlDict*, se recomienda realizar una limpieza de la carpeta de trabajo, para ello existe un script externo con nombre de archivo *Allclean*.

La ejecución del programa se realiza con el comando *sarsa*.

```
***** Simu: 5 *****
epsilon greedy - exploration: 0.384615
survived pedestrian: 1142
Duración: 0 min / 1 s / 1516 ms

***** Simu: 6 *****
epsilon greedy - exploration: 0.333333
survived pedestrian: 1198
Duración: 0 min / 1 s / 1590 ms

***** Simu: 7 *****
epsilon greedy - exploration: 0.294118
survived pedestrian: 1483
Duración: 0 min / 1 s / 1577 ms

***** Simu: 8 *****
epsilon greedy - exploration: 0.263158
survived pedestrian: 1322
Duración: 0 min / 1 s / 1749 ms

***** Simu: 9 *****
epsilon greedy - exploration: 0.238095
survived pedestrian: 1275
Duración: 0 min / 1 s / 1656 ms

***** Simu: 10 *****
epsilon greedy - exploration: 0.217391
survived pedestrian: 1254
Duración: 0 min / 1 s / 1642 ms
```

A su término, se podrá obtener los resultados de cada variable, tablas con información de número de evacuados e imágenes de validación.

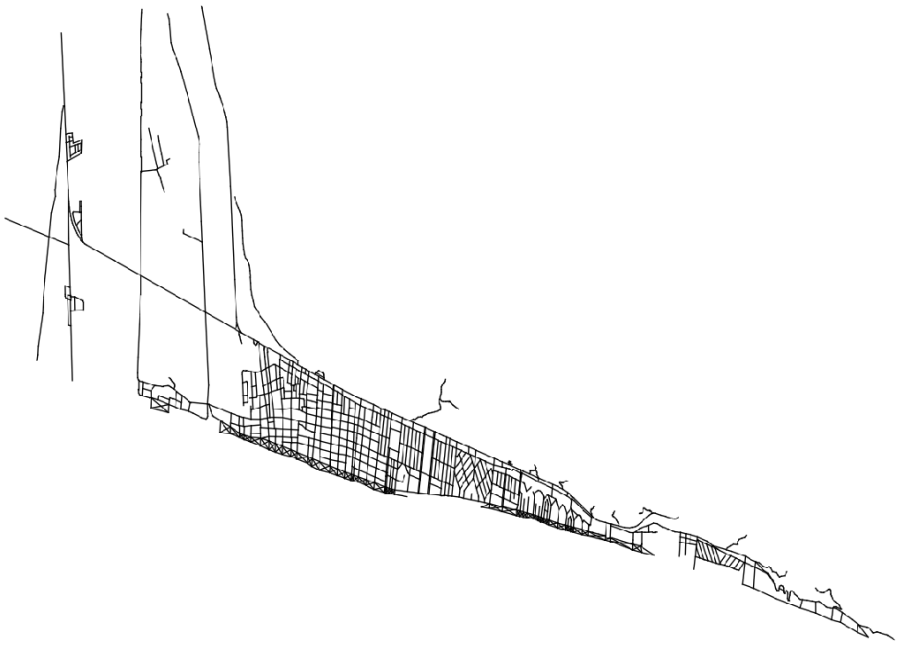
Por último, la exportación de las variables permite al usuario post-procesar la información en cualquier otro programa. A continuación, se muestra el post-procesamiento de la información con el lenguaje python.



6.3.2. Caso Camaná

Este ejemplo mostrará la evacuación de personas en el distrito de Camaná con varias opciones de puntos de evacuación, todas alejadas de la playa, como se muestra en la siguiente figura.

La configuración es una idealización con base en la distribución de calles del distrito de Camaná. Con longitud máxima de calle de 600 metros y un ancho de 3 metros.



La configuración de este caso es el siguiente:

- Tiempo final de evacuación: 800 seg
- Número final de simulaciones: 7000
- Tiempo promedio de salida de persona: 420 seg
- Cantidad de subdivisiones de una calle: 10
- Delta de tiempo: 1 seg
- Periodo de conteo de personas: 1 seg

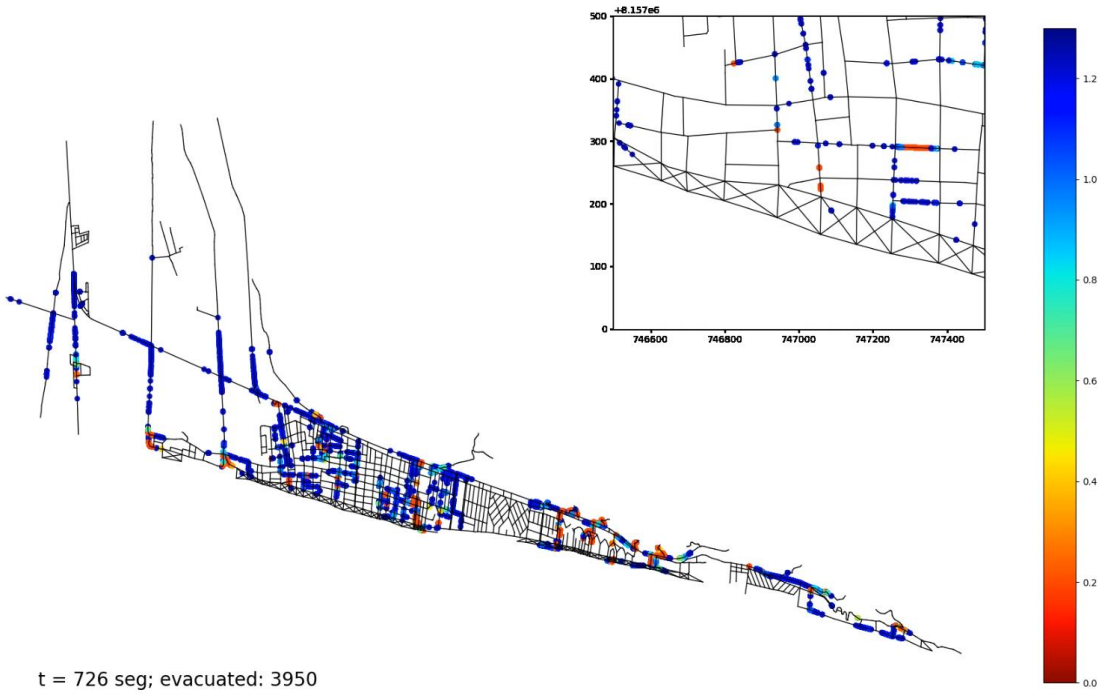
A continuación, se muestra el archivo *controlDict.*

```
nodesFile         nodes.csv;

linksFile         links.csv;

populationsFile   population.csv;

sarsaProcesses trained;

startTime        0;

stopAt           endTime;

endTime          800;

deltaT           1;

meanRayleigh    420;

graphicPrintout no;

graphicPrintoutPeriod 10;

listingPrintoutPeriod 1;

computationContinued    no;

previousComputationFile sim_000000001.csv;

stopSimulationAt          endNumberSimulation;

opcionSubdivision cantidadSubdivisiones;

cantidadSubdivisiones 10;

readPedestrianMassState no;

endNumberSimulation 7000;

numberLinkDivision 10;

evacuatedCount yes;

totalEvacuatedCount yes;

```

Por último, en la carpeta post-procesamiento, existen archivos para realizar visualizaciones de los datos de manera externa al programa principal.

El archivo *snapshot.py*, proporciona imágenes en cada tiempo de paso.

El archivo *zoom.py* permite crear imágenes con una zona amplia para observar a detalle el movimiento de evacuación, como se muestra en la siguiente imagen.



6.3.3. Caso Python

Este caso muestra la manera correcta de importar resultados anteriores de la versión de python de este programa.

```
process calibration;

endTime          800;

meanRayleigh    420;

endNumberSimulation 5;

pythonVersion yes;

pythonOption in;

previousComputationFile pythoncalibrated.csv
```

Se empieza el proceso de calibración de algoritmo con un número de simulaciones de 5, con tiempo de evacuación de 800 segundo cada simulación y un número medio para la distribución de Rayleigh de 420 segundos. Se importan los resultados del archivo *pythonCalibrated*, valores estimados con la versión en python del programa.

Para activar la lectura de resultados de la versión antigua, se debe utilizar el comando *pythonVersion* y activarlo, y el comando *pythonOption* con opción in. El programa utilizará el ordenamiento de python para reordenar a la versión actual, lo cual implica que de allí en adelante los resultados trabajaran con el ordenamiento propicio del programa Escape, por lo que no es necesario activar de nuevo los comando de python.

Luego se puede pasar al proceso *trained* para mostrar resultados del aprendizaje. A continuación, se muestra el archivo de *controlDict.*

```
process trained;

endTime         800;

meanRayleigh   420;

endNumberSimulation 5;

previousComputationFile calibrated.csv
```

7. OTRAS CONFIGURACIONES

En caso se quiera usar datos de *stateMatrix* de simulaciones anteriores, pero de la versión en python del mismo código, se debe activar el comando *pythonVersion*.

A continuación, se muestra el *controlDict* del caso grilla utilizando datos exportados del código en python.

Se debe activar esta opción debido a que el orden de las conexiones de las calles en una intersección es diferente entre los diferentes lenguajes.

8. LISTA DE CLASES

io	Entrada y salida de archivos.
directory	Diccionario de comandos.
node	Definición de intersecciones en una ciudad
link	Definición de calles en una ciudad
sublink	Definición de una subsección de una calle
pedestrian	Definición de peatones
stateMatrix	Estructura de matrix de estados
sarsa	Cómputo de algoritmo sarsa