Optimización de política de evacuación de edificaciones mediante Optimización vía Simulación.

Baquela, Enrique Gabriel*, Porfiri, Diego(1), Ochoa, Joana(2), Olivera Ana Carolina(3)

Grupo de Investigación en Simulación y Optimización Industrial. Facultad Regional San Nicolás, Universidad Tecnológica Nacional.

Colón 332, San Nicolás de los Arroyos, 2900 Bs. As,Argentina.

** ebaquela@frsn.utn.edu.ar*

(1)dporfiri@frsn.utn.edu.ar

(2)jochoa@frsn.utn.edu.ar

(3) Universidad Nacional de la Patagonia Austral.

Ruta N°3 Acceso Norte, 9011, Santa Cruz, Argentina

aolivera@uaco.unpa.edu.ar

RESUMEN.

En este trabajo proponemos una metodología basada en simulación por agentes y optimización mediante recocido simulado para definir políticas de evacuación de edificaciones. La misma consiste en modelar tanto las instalaciones físicas como el flujo de personas en la misma mediante un simulador de agentes de tiempo discreto. Luego, generamos políticas de asignación mediante un algoritmo de recocido simulado y evaluamos la bondad de las mismas en el simulador. Testeamos nuestro modelo sobre las instalaciones de la FRSN-UTN, encontrando muy buenos resultados.

Palabras Claves: Optimización, simulación, agentes discretos, recocido simulado, evacuación.

ABSTRACT

In this paper we propose a methodology based on agent-based simulation and optimization by simulated annealing to define policies evacuation of buildings. It consists in modeling both the physical facilities and the flow of people in a discrete time agent-based simulator. Then, we generate assignation policies through a simulated annealing algorithm and evaluate the goodness of them in the simulator. We tested our model on the building of the UTN FRSN, finding very good results.

1. Introducción

En este trabajo proponemos una metodología basada en simulación de agentes discretos a fin de optimizar y evaluar la evacuación de personas en un edificio. Son muchas las consideraciones a tomar en lo que respecta a evacuaciones: cómo disponer las salidas de emergencias en un edificio, cómo organizar el flujo de personas durante un evacuación, cómo asignar personas a áreas de la edificación de manera tal de evitar congestionamientos en evacuaciones, etc. En el caso de instalaciones ya existentes, en las cuales modificar su diseño no sea fácilmente factible, es interesante determinar

cómo distribuir el personal dentro de las mismas, de manera que el éxito de las evacuaciones sea lo más alto posible. Planteamos aquí un modelo general de simulación de evacuaciones que permite evaluar las configuraciones de los parámetros sobre los cuales se puede ejercer injerencia, y que puede ser vinculado con procedimientos de optimización vía metaheuristicas (recocido simulado en este trabajo) para encontrar configuraciones óptimas.

De esta forma se evaluará cuáles serán las trayectorias de las personas donde pueden darse los cuellos de botellas y así desarrollar medidas preventivas aplicables al escenario de emergencia.

Como caso de aplicación, se seleccionó el edificio de la FRSN-UTN, particularmente su primer piso, modelándose la distribución de personal en un día típico y evaluando la rapidez con la cual es posible su completa evacuación. Luego, se optimizó la distribución de personas en la instalación, atendiendo a restricciones relativas a la conformación de los grupos de personas asociados a una determinada cátedra y la capacidad de los salones para disponerlos.

Para simular el sistema, se utilizó el programa NetLogo, el cual es un entorno de programación orientado a simulaciones basadas en agentes.

2. Formalización del problema

Si bien son varias las métricas a tomar en cuanto a la eficiencia de una evacuación concreta, los dos indicadores más utilizados son la cantidad de personas evacuadas en una determinada ventana de tiempo, y el tiempo en el cual se puede evacuar un determinado porcentaje de la población afectada (idealmente, debería considerarse el 100%). Aunque complementarias (maximizar la primera implica minimizar la segunda), la primer métrica está asociada a la evaluación de evacuaciones ante un suceso puntual, mientras que la segunda es independiente del evento a evaluar. En nuestro trabajo elegimos la segunda métrica. El objetivo de la metodología desarrollada puede entonces expresarse como:

$$\text{Min } Z(x,k) = \text{Max}(\text{tpo\_evacuacion}(i))(x,k) \quad (1)$$

Lo cual implica que el objetivo es encontrar una política “x” de asignación de personas a áreas del establecimiento y una política “k” de asignación de vías de escape que minimice el tiempo de evacuación del individuo “i” que más demore en salir del establecimiento. “tpo_evacuacion” es una función difícil de evaluar analíticamente, por lo cual el enfoque preferido consiste en simular la evacuación y procesar las estadísticas generadas por el simulador.

La variable de decisión “x” es un vector que asigna individuos a áreas del establecimiento. Cada componente del vector representa a un individuo de la población, en el cual su valor es la cave que simboliza al área asignada. Esta asignación no es totalmente libre, ya que la misma debe seguir criterios de conformación de grupos preexistentes (por ejemplo, el conjunto de N personas asignadas en la misma cátedra deben ubicarse en la misma área) así como de capacidad de cada área. Matemáticamente:

Xi = P para todo i perteneciente al grupo I, con p perteneciente al conjunto de áreas P.

Xi <> p para todo i no perteneciente al grupo I, con p perteneciente al conjunto de áreas P.

|i|<= capacidad(p)

La variable de decisión “k” es un vector que asigna, a cada área p, una de las posibles salidas de emergencia.

Para este trabajo, se consideró, que las asignaciones de personas a aulas son fijas. En la dinámica de una jornada estándar, las personas se mueven entre las aulas y las áreas comunes, pero regresan siempre a la misma aula.

3. Desarrollo del Estudio

La primera etapa del estudio realizado consistió en relevar la dinámica de movimientos dentro del establecimiento. Para ello se relevaron la capacidad de cada aula, el cronograma de asignación de

cátedras a aulas y la matrícula de cada cátedra. La información se contrastó con muestreos estadísticos para estimar el flujo de alumnos en la instalación.

En base a las estadísticas relevadas, se confeccionó un modelo de simulación en Netlogo, tomando como sustrato físico el plano del edificio. Se consideraron dos tipos de entidades distintas para modelar el sistema: personas y áreas. Las personas se tipificaron en grupos de alumnos y profesores, asignando a cada grupo una secuencia de movimientos desde área a área en condiciones normales. Las áreas representan el espacio físico de la edificación (salones y espacios comunes). Cada área tiene asociada una capacidad máxima y una ruta de escape fija para todas las personas que estén en el área durante una emergencia.

El modelo de recocido simulado se programó en el propio Netlogo, aprovechando las capacidades del sistema para relevar estadísticas.

Las soluciones potenciales se codificaron mediante un vector, en el cual las primeras N componentes referenciaban a los grupos de personas, y las últimas M componentes a cada una de las áreas. Las componentes de personas tomaron valores comprendidos entre 1 y M, representando el mismo la asignación de grupos a áreas. Las componentes de áreas, por otro lado, toman como valor una de las posibles salidas de emergencia (codificadas en forma numérica).

La solución inicial es generada aleatoriamente, asignando valores por muestreo sin reposición para el caso de las componentes de grupos de personas, y muestro aleatorio con reposición para el caso de las componentes de áreas. El procedimiento de generación de vecinos del enfriamiento simulado consiste en permutación de dos componentes de tipo personas (elegidas aleatoriamente) y la mutación aleatoria por selección para un individuo seleccionado aleatoriamente dentro del conjunto de componente de áreas.

La restricción que impide a más de un grupo ser asignado a un área del tipo aula en simultáneo está abstraída dentro del procedimiento de generación de la solución inicial y el procedimiento de mezcla. El resto de las restricciones se modelaron mediante penalizaciones a la función objetivo, pudiendo una solución tomar cualquier valor dentro del rango de valores permitidos.

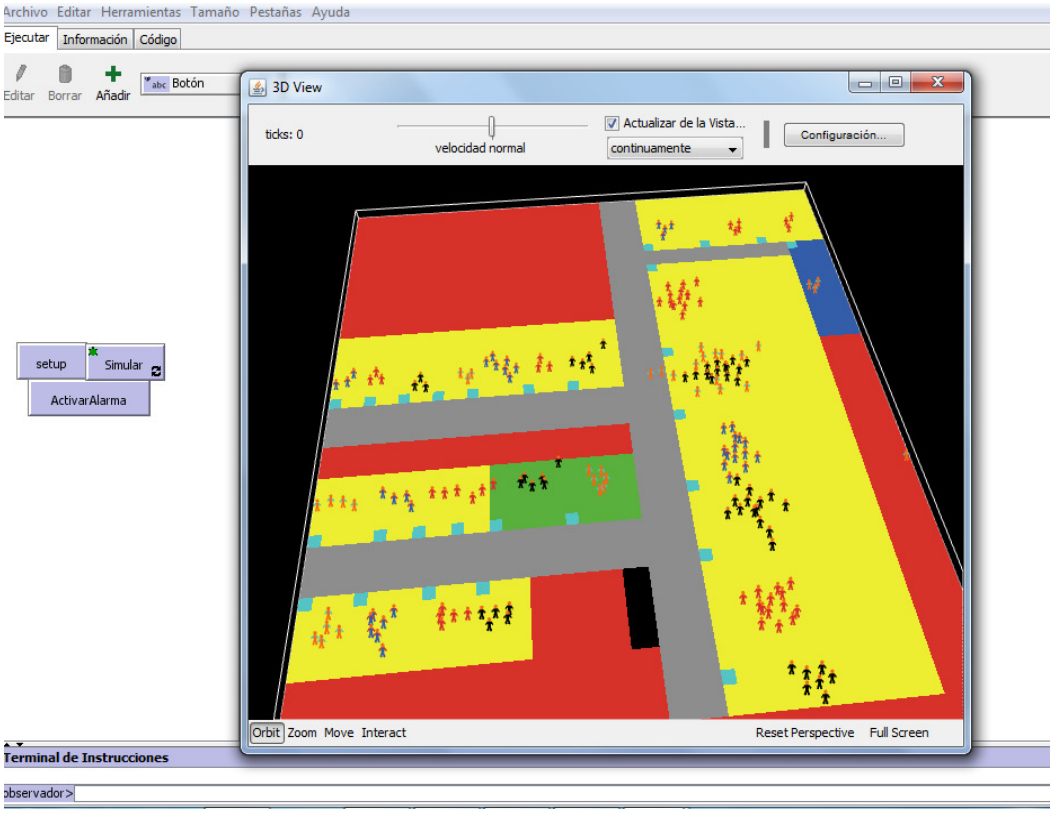


Figura 1 *Visualización de simulación en NetLogo*

4. Resultados Obtenidos

Para testear el modelo se generaron, en primer lugar, asignaciones aleatorias de personal a áreas para evaluar la variabilidad de los tiempos de evacuación, y en segundo lugar, se corrió el algoritmo de enfriamiento simulado. En ambos casos, se simulo el sistema en situación normal y se generó el evento de emergencia en periodos con la mayor parte del personal en aulas y en periodos con la mayor parte del personal en áreas comunes. Tras el evento de evacuación, se consideró que todo el personal se daba por enterado de la emergencia y se dirigía a una salida de emergencia.

No se corrió el algoritmo para configuraciones genéricas de áreas, ni se contempló el caso de cambio de áreas tipo aula para cada grupo de personas, dejándose esas consideraciones para futuros estudios.

Las soluciones generadas por el algoritmo de recocido simulado caen dentro del grupo del decil del 10%, mejorando la asignación vigente de alumnos a salones en un 17%.

Se puede observar que para el caso estudiado, las soluciones generadas por el algoritmo tienen tiempos de evacuación mucho mejores que la media del sistema.

5. Referencias:

Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Botvalde, A. and Lundkvist, M. (2012). Simulation of Building Evacuation, Bachelor thesis, LTH School of Engineering, Lund University (Sweden).

Tzu-Sheng, S. and Shen-Wen, C, (2004), “An evacuation simulation model (ESM) for building evacuation”,International Journal on Architectural Science, Volumen 6, Nro 1, p.15-30, 2005

Baquela E., Gomez L., (2011), “Modelización y simulación de sistemas de evacuación de edificaciones”, Revista EPIO, Nro 31

6. Anexo: Construcción del modelo

6.1 Consideraciones generales

Conociendo en detalle el plano del área a estudiar, se comenzó con la construcción del modelo de optimización.

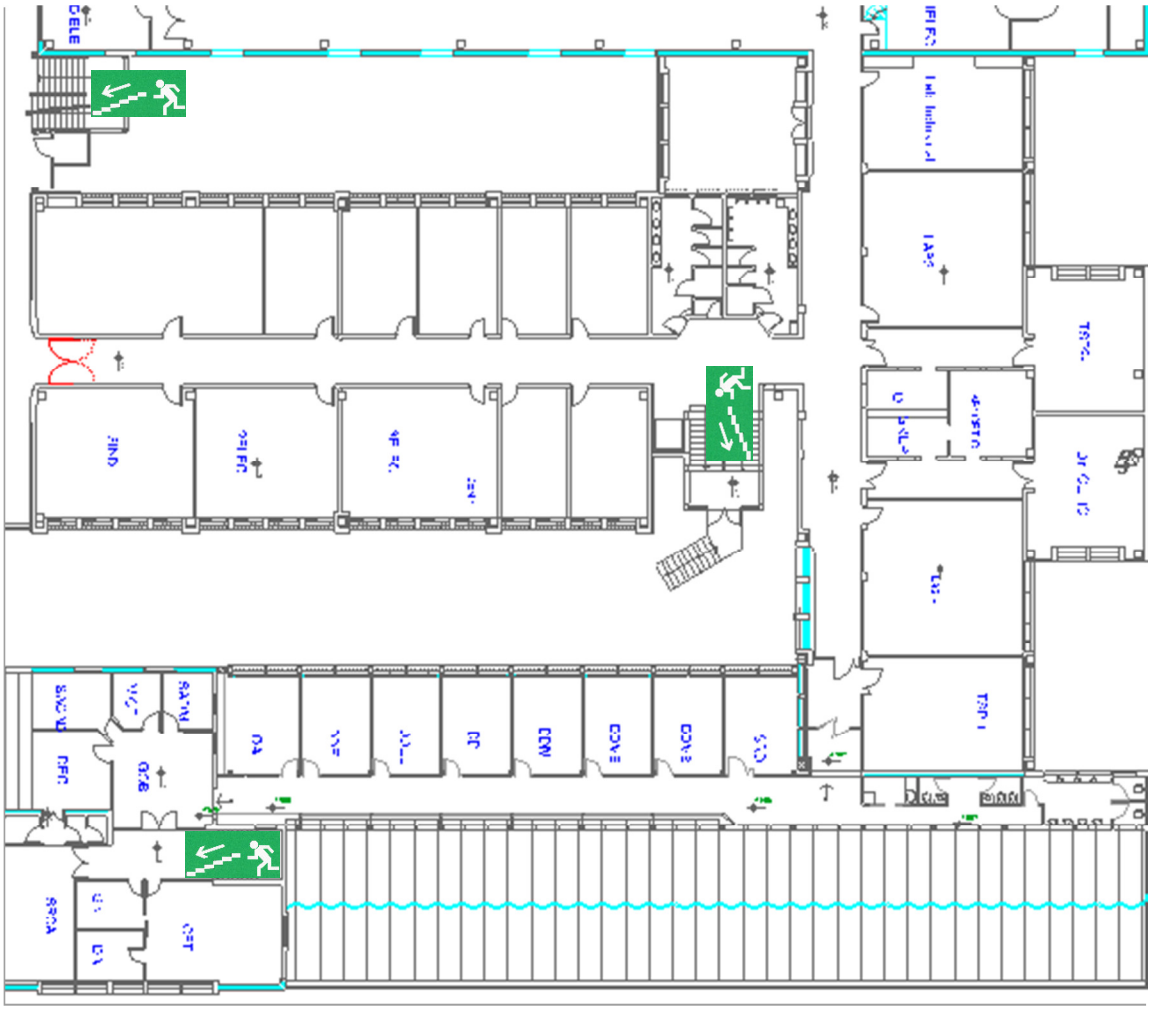


Figura 2 *Plano del 1er Piso, UTN-FRSN*

Para el armado del modelo, se eligió como base la herramienta Netlogo que emplea la simulación mediante agentes.

En Netlogo, se denomina “world” al sustrato en el que se mueven los agentes. Su construcción mediante código es simple e iterativa. Para este trabajo el “world” consta de diferentes salones, diferenciándose estos por la densidad de personas por m2.

Para modelar el sistema, empleamos los siguientes dos tipos de agentes disponibles en Netlogo:

1. Turtles: (tortugas) son los agentes que se mueven por el mundo, interaccionando entre sí y con el medio. Cada tortuga está identificada de forma única, pero puede pertenecer a una “clase”. Se utilizaron para representar personas.

2. Patches: (celdas) es cada porción cuadrada que forma el mundo por las cuales se desplazan las tortugas. Estas también están identificadas, pero lo hacen por las coordenadas de su punto central. Se utilizaron para modelar las áreas.

6.2 Secuencia de armado

Como primer paso se arma el layout del modelo, es decir, se representa el plano definiendo las celdas que pasarán a constituir los pasillos, salones, laboratorios, oficinas y obviamente las salidas de emergencia. Cada uno de estos sectores se identifica por las siguientes propiedades:

- pcolor: color elegido por el programador
- nroSala: número único correspondiente a cada sector en particular
- posXSalida: este valor indica el punto de salida en el eje de las abscisas
- posYSalida: lo mismo que el anterior, salvo que su valor corresponde a las ordenadas

Para ello se definió la siguiente función ,"armarplano", de la cual mostramos un extracto:

```
to armarplano

  if (pxcor > 3) and (pxcor < 6) and (pycor < 30) and (pycor > -6) [set pcolor grey set nroSala "pasillo1" set posXSalida 4 set posYSalida -6]

  if (pxcor > 1) and (pxcor < 6) and (pycor < -5) and (pycor > -24) [set pcolor grey set nroSala "pasillo2" set posXSalida 1 set posYSalida -13]

  if (pxcor > 1) and ( pxcor < 4) and (pycor < 30) and (pycor > 4) [set pcolor grey ]

[…]

End
```

A lo largo de toda esta sentencia sólo se hizo referencia a un solo tipo de agentes, las celdas, a las cuales se les definió las propiedades antes mencionadas.

Para crear las tortugas es necesario asignarles una ubicación inicial específica dándoles valores correspondientes al eje de coordenadas (similar a lo realizado para las celdas). Es así que se crea una cantidad definida de personas para cada salón en función a las muestras obtenidas del estudio previo. A continuación mostramos el extracto de la función. Cada llamada a "*crt N*" crea un conjunto de N tortugas del mismo tipo:

```
to CrearPersonas

crt 14 [

  set shape "person"

  set xcor 8

  set ycor 14

  set velocidad 0.1

  set Ubicacion "salon1"

  CorregirSuperposicion]

[…]

End
```

Del ejemplo anterior la sentencia *Corregirsuperposicion*, nos asegura que al crear todos los alumnos dentro del salón no los posicionará uno encima del otro, sino que lo hará de acuerdo al espacio que le queramos definir entre los mismos.

Con esto ya tendríamos construido el Layout y distribuidos los individuos cada uno en su posición. Podemos ahora definir una nueva variable llamada *Alarma*, que es la que condiciona la forma en la cual actúan los agentes. La misma se caracteriza por ser una variable global y por ser de naturaleza binaria (0 apagada, 1 activada).

Al crear el modelo, *Alarma* estará apagada pasando a su estado de activación a través de la función "*ActivarAlarma*":

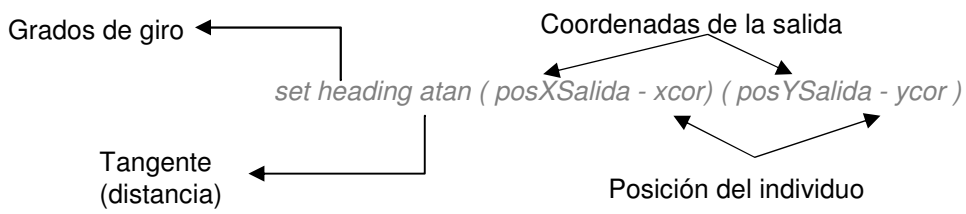
```
to ActivarAlarma
  set Alarma 1
end
```

El bucle que simula el movimiento de los agentes durante el sistema queda entonces encapsulado en la siguiente función:

```
to Simular
  ask turtles with [Ubicacion = "salon1"]
    [if (Alarma = 0) [
     ifelse (nroSala = 1) [
       fd velocidad]
     [
       rt 180
       fd velocidad]
    ]]
    if (Alarma = 1) [
      if nroSala = 1 [
       set heading atan ( posXSalida - xcor) ( posYSalida - ycor )
       fd velocidad]if nroSala = "pasillo1"[set heading atan (posXSalida - xcor) (posYSalida - ycor) fd velocidad]
       if nroSala = "pasillo2" [set heading atan (posXSalida - xcor) (posYSalida - ycor)]]
[…]
end
```

Esta parte del código define el comportamiento de los agentes que tengan como ubicación el "salon1"; donde encontramos, ya sea que esté la alarma apagada o encendida, la sentencia *fd velocidad* que es la encargada de asegurar el movimiento de los agentes a través del valor fijado anteriormente a la variable velocidad.

En el caso de activar el botón Alarma, el recorrido de todos los individuos se altera para dirigirlos hacia la salida de su salón, esto se hace calculando el ángulo que debe girar cada tortuga considerando la diferencia entre su posición y la posición de salida dada por sus coordenadas.



A medida que vayan cambiando de ubicación, los individuos adquirirán distintas direcciones para poder llegar a las salidas de emergencia asignadas.