Tecnológico de Monterrey

Reto: Movilidad Urbana

Modelación de sistemas multiagentes con gráficas computacionales (Gpo 302)

15.10.2023

—

Estudiantes

Gabriel Rodríguez De los Reyes, A01027384

Rodrigo Núñez Magallanes, A01028310

Índice

Problemática

La presente actividad tiene como objetivo abordar la complejidad y dinámicas del tráfico vehicular en entornos urbanos mediante el diseño e implementación de una simulación interactiva. La misma tiene como objetivo entender el comportamiento del tráfico y poder analizar qué cualidades de los conductores son contraproducentes para evitar embotellamientos, encontrando una forma óptima de manejo.

En la simulación de tráfico, utilizaremos la librería Mesa en Python para modelar un escenario urbano que refleje de manera realista las interacciones entre vehículos, destinos, calles y semáforos. Esta simulación se llevará a cabo con un elevado número de agentes, lo que va a contribuir a su comportamiento no determinista con cierto grado de aleatoriedad en sus resultados. La representación gráfica de este entorno se potenciará con Unity 3D, proporcionando una visualización más inmersiva y detallada de los eventos que se desarrollan en la simulación.

En nuestro escenario, se implementarán agentes que representan vehículos autónomos con comportamientos y cualidades similares a aquellas de conductores reales con el fin de obtener los mejores resultados. Se incorporarán semáforos y sistemas viales comunes para introducir elementos de control y desafío adicionales, reflejando así la complejidad del tráfico urbano actual. Cada vehículo autónomo, guiado por algoritmos que priorizan la eficiencia y el camino más corto, contribuirá a la complejidad del entorno urbano en evolución. Factores específicos del agente, como la proactividad, la reactividad, eventos inesperados, la comunicación y la toma de decisiones, se convertirán en elementos cruciales para el éxito de cada vehículo en alcanzar su objetivo

La representación visual en Unity 3D permitirá observar de manera inmersiva cómo los vehículos viajan en la ciudad, adaptándose sus decisiones a las condiciones cambiantes del tráfico y buscando la ruta más acorde hacia sus destinos. A medida que más vehículos se incorporen al escenario, se explorará la dinámica de cómo interactúan entre sí, evitan congestiones y optimizan sus rutas para minimizar el tiempo de viaje. Permitiéndonos entender qué herramientas realmente ayudan al conductor y cuáles resultan ser perjudiciales.

Diseño de los agentes

Tipos de Agente

Dentro de la simulación se pueden encontrar 5 tipos principales de agentes fundamentales para el resultado esperado:

- **Destination**
  - El **Destination** es el agente que funciona para marcar el objetivo al que cada vehículo debe llegar para lograr sus metas. Al entrar en contacto con algún vehículo, el mismo es eliminado de la simulación entendiendo que alcanzó su destino.

  - Ubicación:
    - Viene dada por el símbolo D

  - *No cuenta con ningún método*
  - *No cuenta con ningún atributo*

- **Obstacle**
  - El **Obstacle** es el agente que funciona para delimitar por dónde puede conducir y pasar el vehículo. Dentro de la simulación representa a los edificios y casas en la ciudad, o cualquier lugar donde no hay calle.

  - Ubicación:
    - Viene dada por el símbolo #

  - *No cuenta con ningún método*
  - *No cuenta con ningún atributo*

- **Road**
  - El **Road** es el agente que funciona para definir los sectores por los que puede viajar el vehículo y el sentido en el que lo hace. Dentro de la simulación cumplen el rol de las calles con su respectivo sentido

  - Ubicación:
    - Viene dada por el símbolo <, >, ^, v

  - *No cuenta con ningún método*
  - *No cuenta con ningún atributo*

- **Traffic_Light**
  - El **Traffic Light** es el agente que simula los semáforos dentro de toda la simulación, y puede permitir o limitar el pase de los vehículos por un punto concreto dependiendo de su estado

  - Ubicación:
    - Viene dada por el símbolo *s* o *S*

  - Atributos
    - state
      - Estado actual, {prendido: verde, apagado: rojo}
    - timeToChange
      - Tiempo que pasa en cada estado

  - *No cuenta con ningún atributo*

- **Car**
  - El **Car** es el agente que simula el vehículo dentro de la simulación, su objetivo es lograr llegar a su destino de la forma más rápida posible sin colisionar ni generar bloqueo en el tráfico.

  - Ubicación:
    - Aparece en agente en cada esquina del mapa cada 10 *steps*

  - Atributos
    - Goal
      - Destino objetivo del carro
    - patience
      - Paciencia del vehículo valor aleatorio del 5, 10
    - map
      - Grafo de la ciudad
    - path
      - Ruta al destino objetivo calculada con **A***
  - Métodos
    - calculate_A_star(start_point, destination_point)
      - Calcula el mejor camino del punto *A* al *B* can **A***
    - move()
      - Recorre el camino encontrado por *calculate_A_star*
    - Step()
      - Controla lo que se ejecuta en cada iteración
    - out_of_patience()
      - Cambia los pesos del grafo, y recalcula el camino

Arquitectura de subsunción de “Car”

Arquitectura de subsunción			
	Capa	Objetivo	Comportamiento
1	Siguiente posición está libre	Continuar recorrido	El agente avanza a la siguiente posición de su recorrido
2	Hay semáforo verde	Avanzar	El agente avanza a la siguiente posición de su recorrido
3	Quiere cruzar, pero tiene un coche a su derecha	Dejarlo pasar	Espera un turno para dejarlo pasar y después cruza
4	Hay semáforo rojo	Esperar	No avanza ni reduce su paciencia
5	Hay coche en su siguiente posición	Se reduce su paciencia y no se mueve	No avanza y reduce su paciencia
6	Ya no tiene paciencia	Busca un nuevo camino	Aumenta el peso del vértice y recalcula el renuevo recorrido con **A***
7	Llegó a su destino	Desaparecer	Se elimina del agente del tablero y de la simulación

Tabla 1: Arquitectura de subsunción

Análisis de los agentes

Objetivos

El objetivo del agente **Car** consta de llegar de un punto A hasta un punto de la manera más corta posible. Cabe destacar que para lograr esto el mismo tiene ciertos objetivos que lo ayudan a alcanzar sus metas. Entre estos se destaca, que el vehículo tiene el objetivo de tratar de continuar moviéndose para evitar embotellamientos, por lo que siempre intentará seguir avanzando por donde pueda. Otro subjetivo, sería priorizar viajes en línea recta sobre las diagonales, lo que facilita su interacción con otros conductores, y en general mejora su probabilidad de llegar al destino.

Percepción y Sensores

Dentro del agente principal podemos encontrar ciertas propiedades que le permiten interactuar con su entorno de forma proactiva y reactiva. El agente tiene la capacidad de conocer los vehículos que se encuentran a su alrededor, al igual que el estado en el que se encuentran. Estos sensores le permiten recolectar información que utiliza en la toma de decisiones para llegar a su objetivo. Esto sé ve reflejado en escenarios donde el agente se encuentra otro vehículo detenido, que dependiendo sus índices de paciencia, decidirá esquivarlo o simplemente esperar un poco.

Estos sensores no solo le permiten conocer la situación del tráfico, ya que también cumplen un rol fundamental en la comunicación con otros agentes. En el caso de los semáforos, el agente utiliza sus sensores para conocer el estado actual del mismo y conocer si puede continuar o debe detenerse. Debido a que los semáforos conllevan una pausa adicional a nuestros agentes, los mismos suelen tratar de evitarlos en la medida de lo posible, si tiene una alternativa de costo similar.

Todos estos sensores le permiten al agente reaccionar a diferentes situaciones que ocurren en la simulación, pero su percepción del espacio le permite ser proactivo hacia su objetivo final. Pese a que los agentes no poseen cualidades conscientes sobre la ciudad, ya que no conocen la posición de otros agentes, ni el estado de todos los semáforos, los mismos sí poseen el conocimiento total de las calles. Cada agente tiene una representación en formato de grafo de las calles de la ciudad que le permiten calcular el mejor recorrido desde su posición hasta el destino final. Dentro de este, el agente define sus pesos con lo previamente mencionado, priorizando sus calles rectas a aquellas diagonales.

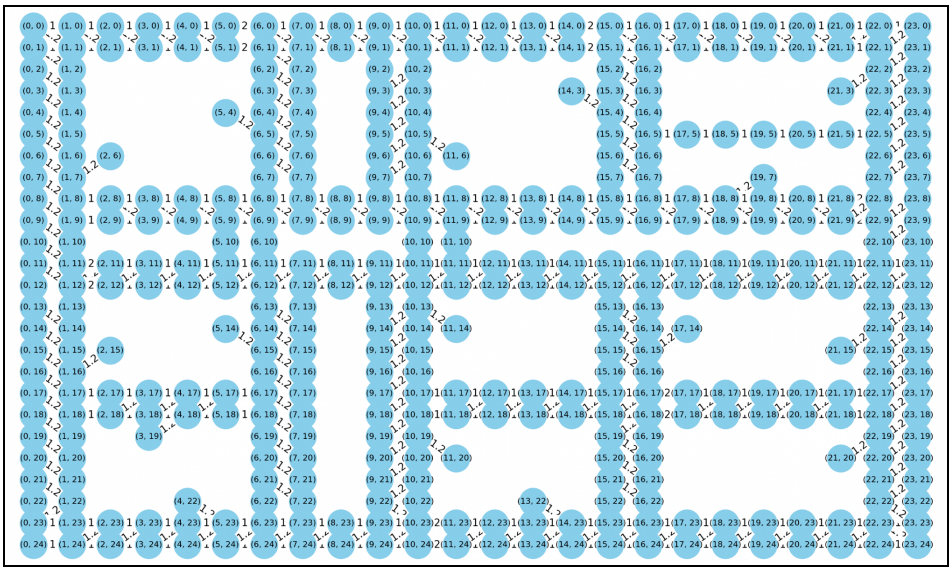

Imagen 1: Grafo de la ciudad

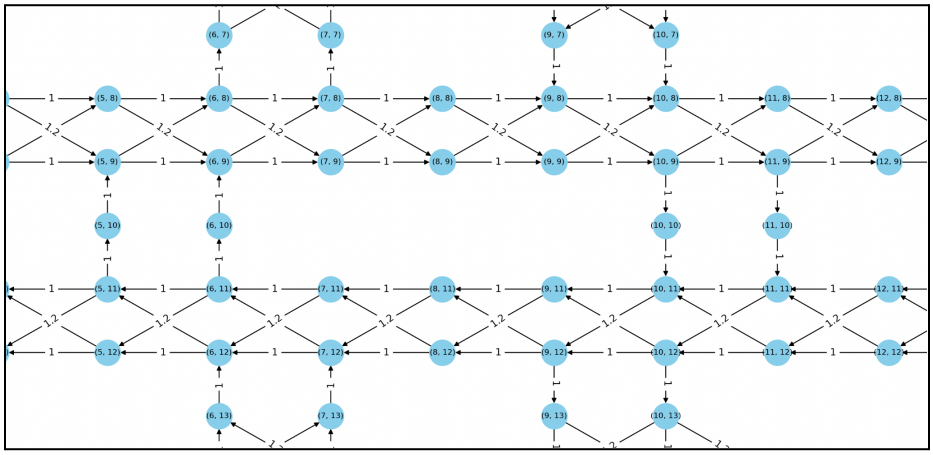

Imagen 2: Grafo de la ciudad ampliado

Capacidad efectora y proactividad

La proactividad y la capacidad efectiva de nuestros agentes fue una parte fundamental de la implementación para evitar el tráfico y los embotellamientos. Nuestros agentes cumplen la capacidad de proactividad al siempre estar en búsqueda de su objetivo, en este caso, llegar a su destino. La primera acción que ejecuta el agente al momento de entrar en la escena es calcular su A* para poder conocer en qué dirección empezar a moverse. El mismo siempre seguirá esta ruta al menos que se encuentre un obstáculo, otro agente o un semáforo rojo delante. Durante estos escenarios, el agente debe ser reactivo para evitar colisiones o simplemente saliéndose del mapa.

Pese a que en todo estos casos la reacción inmediata sea frenar para evitar el impacto, dentro de la simulación cada uno se los conlleva un resultado diferente. Si el agente se frenó debido a estar en un semáforo en rojo, el mismo no gastará su paciencia ni modificará su ruta

original al entender que esta situación está fuera de su control y no puede cambiarlo. Ahora, si la falta de movimiento se debe a tener vehículos delante de él, su rumbo de acción será diferente. En ese escenario, el agente aumentará el peso de su camino actual y recalculará su A*, buscando una ruta que sea más favorecedora. Esto le permite al agente reaccionar ante estos escenarios, pero siempre tratando de llevarlo a su objetivo, que consiste en llegar a su destino.

Este cambio de dirección no sucede cada vez que se encuentra otro vehículo, ya que esto no permitía una mejora real. Por ende, se decidió tener una métrica de paciencia que bajo de forma progresiva cuando el vehículo no se puede mover por culpa de otro. En el caso de llegar a 0 se ejecuta la acción mencionada anteriormente, y el agente recarga su paciencia hasta encontrar otro escenario similar.

Por último, con fines de facilitar la interacción entre nuestros agentes, decidimos definir una regla de prioridad en los cruces. En donde dos vehículos no podrían ejecutar un cruce a su diagonal opuesta durante el mismo *step*. Este se debe a que pese a ser un cruce teóricamente permitido en mesa dentro de la simulación de *Unity* este movimiento parecía innatural, ya que los carros parecían fusionarse. Para esto se decidió, que en el caso de que dos vehículos estén buscando ejecutar esta diagonal de forma opuesta, el que se encuentre en el carril derecho tendrá la prioridad sobre el otro. Esta dirección es relativa al sentido de la calle, por lo que los agentes deben conocer su sentido para comunicarse con los demás agentes y darle prioridad a aquellos del canal de derecho, siempre y cuando no se queden si paciencia.

Métricas de desempeño

Para medir el desempeño de los agentes dentro de la simulación se podrían elegir ciertos atributos que ayudarían a juzgar el comportamiento. En principio, uno de estos sería cuántos vehículos activos se encuentran en la simulación; pero este no nos resultó de tanta ayuda, ya que se veía favorecido por las congestiones. Debido a esto se decidió usar una medida diferente basándose en cuántos vehículos lograban llegar a su destino en un tiempo determinado. Para hacer esto se definió que se analizaría como era el comportamiento de los agentes, en término de llegar a su destino, dentro de los primeros 1000 pasos. Esto se llevó a cabo incrementando un contador cada vez que un vehículo alcanzara su destino.

Con el fin de poder entender el comportamiento de nuestros agentes en múltiples escenarios, se añadieron ciertos controladores que nos permitían modificar la interacción de nuestros agentes. El primer atributo era el *respeto*, este atributo consiste en que tanto el agente debía conducir siguiendo buenas prácticas, o simplemente podría hacer lo que quisiera. A nivel interno del código, este atributo modifica el valor de las diagonales restringiendo que tanto se podría cambiar de canal. El segundo atributo, la *paciencia* es el parámetro encargado de designar que tanto está dispuesto el agente a quedarse quieto esperando. Mientras este valor se incremente, el agente pasará más tiempo esperando antes de modificar su ruta. A nivel interno, este parámetro modifica directamente la paciencia del vehículo, que al llegar a 0 cambia los pesos del grafo y calcula su ruta.

Por último, el tercer atributo consiste en una métrica que designa el costo que supone un semáforo para nuestros agentes. Mientras este valor sea mayor, los agentes preferirán tomar otras rutas que no involucren utilizar semáforos, al menos que sea estrictamente necesario. Dicho parámetro tiene la capacidad de modificar el grafo original alterando el costo de los vértices de los semáforos.

Imagen 3: Imagen de ajustadores de meta

Para medir el desempeño de nuestros agentes decidimos alternar estos parámetros para así poder elegir la mejor configuración para ellos. Dentro del análisis de resultado se simuló cada escenario 10 veces y se tomó su mejor iteración.

Escenario 1 (Respeto bajo, Paciencia baja, Energía baja)

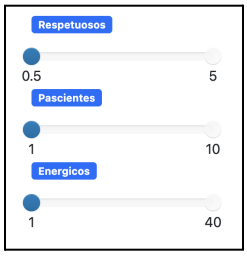

Imagen 4: Imagen de ajustadores de meta escenario 1

Dado los atributos proporcionados al agente en esta simulación, el agente manejaba de forma extremadamente agresiva sin respetar a los otros conductores ni intentar seguir estándares de conducción. Su único parámetro de control era que en este escenario el coste de esperar en el semáforo no era muy bajo. Pese a que al ver la simulación la conducción claramente era errática y no simulaba una conducción real, los resultados fueron mejor de lo esperado. Esto en parte creemos que se debió a que al estar moviéndose de forma tan rápida (siempre evitando colisiones), su llegada a los destinos era de forma muy elevada. Aunque siempre al acercarnos a la iteración 1000, el modelo empezaba a generar congestiones, y en general no parecía una representación muy realista del tráfico.

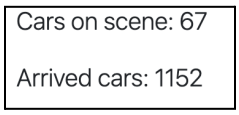

Imagen 5: Imagen de resultados de meta escenario 1

Escenario 2 (Respeto alto, Paciencia alta, Energía alta)

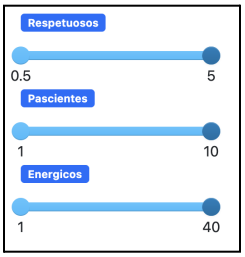

Imagen 6: Imagen de ajustadores de meta escenario 2

Dentro de este segundo escenario se le proporcionaron al agente las instrucciones opuestas al anterior. Priorizando una conducción calmada y respetuosa con los demás conductores, aunque prefiriendo no utilizar rutas con semáforos. Al ver esta simulación se vio una conducción mucho más similar a aquella de la vida cotidiana, donde los automóviles no se cambiaban de canal de forma completamente arbitraría, o sin sentido. Pese a esto, y para la sorpresa de nosotros, este mecanismo obtuvo resultados muy inferiores al anterior, cayendo un

13% en su capacidad de llegar al destino debido al gran número de pequeños embotellamientos que surgieron por falta de productividad y reactividad. Resultado que, pese a parecer poco, puede tener un gran impacto en el tránsito de una ciudad.

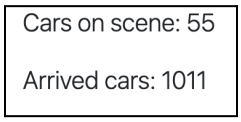

Imagen 7: Imagen de resultados de meta escenario 2

Escenario 3 (Respeto medio, Paciencia media, Energía baja)

Imagen 8: Imagen de ajustadores de meta escenario 3

Con todo lo aprendido de lo anterior se construyó el que idealmente sería un casi ideal para los vehículos. Donde, sin tener demasiada paciencia, ni ser irrespetuosos con los demás conductores, el agente llega a su destino. Este escenario sí mostró los mejores resultados y fue el más constante, siempre obteniendo marcas similares. En este escenario se puedo apreciar una conducción mucho más realista sin los constantes atascos, lo que mejoraba de gran manera el flujo. En ningún momento de las iteraciones se vio en riesgo la simulación con un embotellamiento, mostrando así los mejores resultados encontrados hasta ahora.

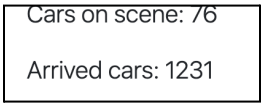

Imagen 9: Imagen de resultados de meta escenario 3

Escenario Extra (Respeto medio, Paciencia media, Energía Alta)

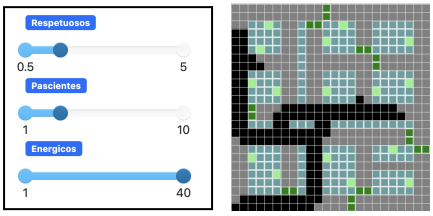

Imagen 10: Imagen de ajustadores de meta escenario extra

Este último escenario mostró los resultados más curiosos, ya que al copiar las condiciones anteriores, pero aumentando el degradado hacia los semáforos, nos encontramos con este caso. En un principio tuvo el mayor crecimiento, obteniendo casi 300 vehículos en sus primeros 200, por lo que parece que habíamos encontrado el escenario ideal. Esto se arruinó cuando alrededor del paso 600 - 800, siempre generaba un embotellamiento total. Esto nos hizo concluir que el comportamiento de los agentes se ve beneficiado por la falta de semáforos en poblaciones reducidas, pero al incrementar esta los agentes empezar a tomar rutas menos eficientes que perjudicaron a la simulación.

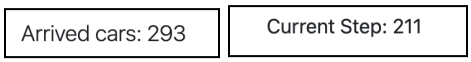

Imagen 11: Imagen de resultados de meta escenario extra

Características del ambiente

El ambiente en el que se desarrollan nuestros agentes es parcialmente accesible, ya que mientras los agentes no pueden obtener información completa de todo el ambiente, sí pueden obtener cierta información. Entre esta información se encuentra cuál es la ruta más rápida a su destino, los estados de los coches que se encuentran en frente y al lado de él, y el estado del semáforo que se encuentra cerca de él. Un poco más de accesibilidad, como la posibilidad de ver 3 coches hacia adelante, por ejemplo, tal vez permitiría al agente tomar mejores decisiones de cambio de carril. No es determinista porque aunque cada acción del agente garantiza un resultado sobre el estado del mismo, el estado del ambiente se ve afectado de una forma “impredecible”, ya que cada acción del agente tiene la posibilidad de cambiar las decisiones de los otros agentes.

El ambiente es episódico, ya que el agente no razona sobre las implicaciones de tomar ciertas decisiones, así como que solo toma decisiones basándose en la información del episodio actual. Dada la presencia de semáforos que cambian de color y controlan el avance de los agentes en esas posiciones, el ambiente toma cierto grado de dinamismo, pues el cambio de color de los semáforos sale del control de los agentes. Algo que también hace al ambiente dinámico son las mismas tomas de decisiones de otros agentes, pues hay casos en donde el agente espera a que su agente dé al lado, se incorpore a su carril, por ejemplo. Si nadie se estuviera queriendo incorporar a mi carril, entonces seguiría avanzando, pero el hecho de que haya otros coches en la calle influye en qué decisiones tomó para moverme. Finalmente, se trata de un ambiente discreto, pues las acciones que pueden realizar los agentes son fijas y finitas.

En otras palabras, como ya se sabe, el ambiente contiene otros coches que toman decisiones, además de semáforos. Algunos coches tienen prioridad de movimiento (como en el caso mencionado del cambio de carril), así como sucede en la vida real. Quizás el modelo no es óptimo en este sentido, y algo que podría mejorar sería la accesibilidad al ambiente. No solo para ver cuántos coches se encuentran en mi propia calle y ver si me cambio de carril o no, sino también para saber qué calles se encuentran congestionadas; y ver si cambiando de ruta puede ser un camino más corto. Porque, actualmente, aunque una calle de mi posible camino está congestionada, si es parte de mi ruta, la voy a tomar, y eso no es “inteligente”.

Visualización

En términos de la visualización, la escena de Unity creaba la ciudad a partir del archivo de texto proporcionado, utilizando una variedad de cuatro Prefabs diferentes para los edificios, y cinco para los autos (que en realidad son el mismo, pero solo cambian de color), para dar cierta variedad y aleatoriedad a la ciudad, aproximándolo a algo más realista. Los semáforos utilizan luces de punto y cambian según la información que reciben de la simulación de mesa a través del servidor de flash. Todos los movimientos de los autos son realizados utilizando las

matrices de transformación 4x4 vistas en el curso, así como interpolación lineal para aparentar el avance, en lugar de utilizar las transformaciones nativas de Unity.

Conclusiones

Este proyecto permitió integrar nuevos elementos y conocimientos de software y de computación en general. Decididamente, si no hubiera habido las actividades previas de simulaciones y de gráficas, este proyecto hubiera tomado las 5 semanas del curso. Por un lado, al aplicar matrices de transformación en Unity en lugar de usar las transformaciones “nativas” agregó un grado de complejidad al proyecto, pero permitió comprender cómo funcionan estos movimientos de los objetos en la escena, donde quizás lo más complicado fue la parte de rotar los autos a sus direcciones designadas. Saber utilizar este software es muy útil y abre muchas posibilidades dentro del mundo de la computación y de la creatividad. En general, comprender cómo funciona un poco detrás de escena, la renderización y animación de los objetos también fue enriquecedor.

Por otro lado, cabe mencionar que la parte de agentes fue interesante y sin duda pone a pensar sobre qué tanto podemos aproximar el mundo real a partir de los sistemas computacionales, y es un prospecto interesante y emocionante. Simular esto, por ejemplo, con un ambiente continuo, sería muy complejo, pero podría ser mucho más cercano a la realidad, y es muy interesante cómo la arquitectura de subsunción de los agentes permite entender sus prioridades y su proceso de toma de decisiones.

Por último, este proyecto es una clara demostración del poder de los agentes computacionales y su aplicación en el mundo real. Pese a ser una simulación muy simple, se lograron extrapolar algunos resultados al mundo real, entendiendo muchas interacciones reales atreves de un modelo simulado. Pese a que esta sería una representación del problema relajado, dado que no contempla una innombrable cantidad de condiciones del mundo real, sistemas como estos aíslan comportamiento permitiendo su análisis. De igual manera, el desarrollo demostró que mientras más sencillas y fundamentales sean aquellas reglas de nuestros agentes y del modelo, mejor serán los resultados y más libertad nos darán de ver como reaccionan y aprender de su comportamiento.