Informe del Proyecto Sea Channel. Simulación. Curso 2020-2021

Leonel Alejandro García López L.GARCIA3@ESTUDIANTES.MATCOM.UH.CU
Grupo C412

Tutor(es):
Dr. Yudivián Almeida Cruz, *Facultad de Matemática y Computación, Universidad de La Habana*
Lic. Gabriela Rodriguez Santa Cruz Pacheco, *Facultad de Matemática y Computación, Universidad de La Habana*
Lic. Daniel Alejandro Valdés Pérez, *Facultad de Matemática y Computación, Universidad de La Habana*

Tema: Simulación, Eventos Discretos

1. Introducción

Como proyecto se propuso simular el funcionamiento de un canal marítimo y a partir de este calcular un tiempo de espera estimado de los barcos, para lo cual asumimos como tiempo de espera para un barco en la cola de arribos a un dique como el tiempo que demora el dique en registrarlo para entrar (proceso de registro inmediatamente previo a la apertura de la reclusa de entrada al dique). Encontré similitud con una arquitectura de 5 servidores conectados en serie, y a partir de ahí comenzó la implementación, donde los servidores serían diques y los clientes las naves.

1.1 Modelado

El modelo empleado en la simulación, posee las entidades básicas 'Ship' y 'Dike', que representan lo evidente pero solo funcionan como contenedores, u objetos de registro para almacenar información de lo que representan pero no del tiempo de la simulación actual. Para manejar todo cronológicamente utilizando una cola con prioridad de 'Event's que facilitaba el acceso al próximo evento en tiempo logarítmico, los eventos muestro a continuación:

Event → Evento abstracto y general, que posee un *time* y una $PRIORITY$ (prioridad sobre otros eventos si que ocurren simultaneamente).

ShipArriveToChannel → Evento que describe un arribo de nave al canal, contiene la propiedad *ship* asociada a la misma.

EnqueueToDike → Evento que contiene una nave *ship* y el *dike* siguiente en el canal al que la nave espera para entrar.

OpenDikeEntryGates → Evento que describe el inicio de la apertura de las puertas de entrada al *dike* y registra a los barcos que se encuentran en la cola de arribos y caben dentro del *dike*, para su posterior entrada (a partir de este punto las naves ya no están esperando a ser atendidas ya están en proceso de entrada).

TransportShipsInsideDike → Evento que describe el evento de entrada al *dike* de las naves que fueron registradas previamente en el manejo del evento *OpenDikeEntryGates* (secuencial, dependiente del tiempo de los arribos).

TransportThroughDike → Evento que da inicio a la fase de transporte dentro de un *dike*.

ExitShipsFromDike → Evento que da inicio a la fase de salida de un *dike*.

ExitShipsFromChannel → Evento que denota la salida de un grupo de naves *ships* del Canal Marítimo

Para el manejo de eventos se implementó el $SeaChannelEventsLoopHandler$ que contiene el estado actual del Canal en sus campos, y maneja el flujo de eventos así como la creación de estos. Son almacenados en una cola con prioridad, donde a menor tiempo mejor posicionados están para su extracción y manejo (en caso de ocurrir simultáneos se consulta la prioridad del tipo de evento). En este controlador es donde se maneja el tiempo de la simulación, se lleva un registro de todos los barcos que culminaron la simulación, así como de la cola de eventos y la lista de diques.

1.2 Puntos Ambiguos y Convenios

Para discretizar procesos de la vida real siempre se debe tener en cuenta que tan compleja debemos realizar la simulación, este problema no es un caso aislado, primero para esclarecer el tiempo de espera, solo tome en cuenta el desde el momento de llegada a la cola de arribos en cada dique hasta el registro en estos para su entrada. En la fase de salida también hay un punto por aclarar, donde los barcos que están en el interior del dique salen todos con un tiempo de salida igual, así que el arribo al siguiente dique es simultaneo para estas naves. Por otro lado, la creación de algunos eventos como el $ShipArriveToChannel$ y $ExitShipsFromChannel$ aunque parezcan innecesarios son para mantener la lógica y el flujo de eventos intuitivo, simple de entender y extender.

2. Simulación

El flujo de la simulación es como sigue, primero se construyen unas tres listas de arribos para cada uno de los 3 tamaños (de barcos) definidos en el problema, luego

se construye la cola de eventos inicial con los arribo de estas 3 listas. En este punto se inicia el ciclo principal, donde se manejan los eventos en orden cronológico, mientras la cola este por completo vacía. Llegados a este punto es bastante simple inferir que se realiza para manejar cada evento y que eventos desencadena cada uno. En el caso del *ShipArriveToChannel* solo encola un evento de tipo *EnqueueToDike* referido a la nave en cuestión y al primer dique.

El *EnqueueToDike* como condición evita que una vez alcanzado el tiempo final se permita el paso al primer dique, dado que solo esta permitida la salida del canal o procesos para esta, llegados a este tiempo. Este evento añade a la cola del dique siguiente el barco en cuestión y encola un *OpenDikeEntryGates* para abrir el dique en caso de que esté en desuso.

El *OpenDikeEntryGates* que llama a abrir las puertas en un tiempo determinado que distribuye exponencial($\lambda = 4$minutos), registra las embarcaciones que caben dentro del dique antes de abrir las compuertas (una especie de reservación), donde se asigna el tiempo de espera para cada una de las naves, como la diferencia del tiempo de registrado actual (previo a la apertura de compuertas) y el arribo a la cola del dique. Finalmente lanza un evento para transportar los barcos al interior del dique el $TransportShipsInsideDike$.

El $TransportShipsInsideDike$ un aviso de que hay barcos registrados para entrar a un dique determinado y estos van entrando de manera secuencial de acuerdo a la cola de arribos(mismo orden en que fueron registrados), donde cada barco demora posicionándose en el interior un tiempo que distribuye exponencial($\lambda =$ 2minutos) independiente de su tamaño, y se realiza un llamado a transportar el grupo de naves en el interior, digámosle flota.

El $TransportThroughDike$ es lanzado para transportar la flota en el interior a la segunda parte del dique, en un tiempo que distribuye exponencial($\lambda =$ 7minutos), a su vez encola un evento de tipo $ExitShipsFromDike$, con esta misma flota.

El $ExitShipsFromDike$ es encolado para remover o definir el proceso de salida, que depende de la número de naves que componen la flota que se traslada en el interior, que distribuye exponencial($\lambda = 3/2$minutos)por la cantidad de barcos que saldrán (no secuencial) donde todas las naves salen de este proceso con un tiempo de arribo al siguiente dique simultaneo llamando a *EnqueueToDike* con cada uno de los barcos que salieron, en caso de ser el último dique se realiza un aviso del tipo $ExitShipsFromChannel$, y luego se restaura el estado de dique anterior, haciéndose un llamado a *OpenDikeEntryGates* como señal de que se culminó la tarea que mantenía el dique ocupado (en caso de que en su cola estén esperando por continuar).

El $ExitShipsFromChannel$ por último es lanzado para almacenar los barcos que salen del canal y sacar estadísticas de ellos al finalizar la simulación.

2.1 Resultados Obtenidos

Para un total de 250 barcos como máximo, los resultados de *tiempos de espera* para todos los barcos en total fueron de entre $[220, 398]$ minutos, para cada barco que entra al dique un máximo de 1.7 minutos y un mínimo de 0.50 minutos aproximadamente, a continuación algunos ejemplos:

Total Ships:	247
Total Awaited:	274.39267537714954
Median Awaited:	1.1109015197455447

Total Ships:	246
Total Awaited:	319.00497020011363
Median Awaited:	1.2967681715451773

Total Ships:	244
Total Awaited:	321.1813121762692
Median Awaited:	1.3163168531814313