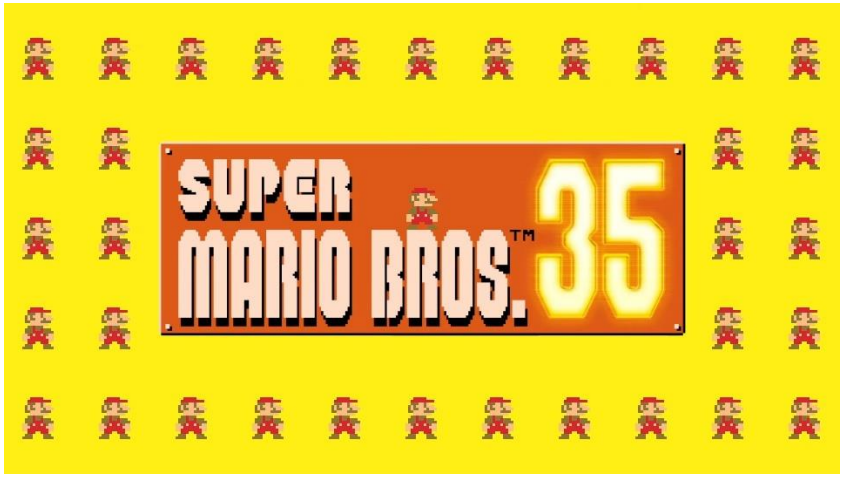
Universidad de Costa Rica

Escuela de Ciencias de la Computación e informática

CI0117- Programación Paralela y Concurrente

Manual de Usuario para el programa:

"Proyecto 2: Simulador de Battle Royale de Procesos [super_mario_mpi]"



Estudiantes:

Gianfranco Bagnarello Hernández B70866

Katherine González Arias B22867

Profesor: Jose Andrés Mena Arias

II Semestre, diciembre 2020

Objetivo

El objetivo de este proyecto es lograr la correcta implementación de un simulador de un Battle Royale de Procesos utilizando la interfaz de paso de mensajes o MPI (sigla del inglés message passing interface) que nos permita recrear una versión muy simplificada del juego Super Mario Bross 35, donde hay varios procesos simultáneamente simulando el juego e interactuando entre ellos. Para cada proceso, un jugador virtual simulará el recorrido de Mario a lo largo del primer mundo (1-1) de su versión original.

Bibliotecas externa utilizada en este proyecto:

Para lograr la correcta ejecución de la simulación, le recomendamos instalar los siguientes comandos en su máquina Linux para de esta forma tener lista la librería de MPI:

sudo apt-get install -y mpich

Compilar el código fuente.

Para la compilar de manera correcta del código fuente y la generación del archivo ejecutable se creó un *Makefile* con el siguiente código:

```
compile: main.cpp

	g++ -Wall -o super-mario-mpi main.cpp
```

Opciones del programa

1. Ingreso a la simulación

Para ingresar a la simulación, debe llamar a *make* desde consola y una vez compilado el programa debe escribir la siguiente directiva:

mpiexec -n 7 ./super-mario-mpi 2 R

Donde cada uno de estos valores representa:

mpiexec: es el llamada al ejecutable de un programa que utiliza directivas de MPI.

-n 7: especifica el número de procesos que usa, en este caso hay 7.

./super-mario-mpi: especifica la ruta del programa a ejecutar.

2: representa en numero del proceso al que se quiere ver jugar.

R: es la estrategia que se elige para atacar a los contrincantes. En este caso es R que representa random, pero podría ser L para acatar a quien tiene menos mones, M para atacar a quien tiene más monedas y A para que un jugador A ataque a uno de los jugadores atacantes cada vez que elimina un enemigo.

2. Inicialización de la partida y comportamiento del programa

Una vez inicializada la partida, un Mario empezará a recorrer el Mundo 1-1, en el cual se enfrentará a varios elementos, dentro de los cuales

habrá enemigos del tipo little goomba o koopa tropa, obstáculos del tipo hole, además de compensaciones del tipo coin.

Una vez Mario se encuentre frente a un enemigo, ya sea little goomba o koopa tropa, hará una decisión basada en tres opciones:

- *Mario no salta. Mario muere y el juego termina (Game Over).*
- *Mario salta y pasa.*
- *Mario salta y mata al enemigo.*

Cuando Mario se enfrente a un *hole* podrá tomar las siguientes decisiones:

- *Mario no salta. Mario cae y el juego termina (Game Over).*
- *Mario salta y pasa.*

Finalmente, cuando Mario se encuentre con un *coin* podrá tomar alguna de las siguientes decisiones:

- *Mario no salta. Esta acción no tiene ningún efecto.*
- *Mario salta y golpea el bloque. Mario obtiene una moneda.*

Mientras Mario se desplaza por el mundo, paralelamente habrá otros Marios jugando la misma partida en sus respectivos mundos, a los cuales, mediante las estrategias de ataque de R, L, M o A, descritas previamente puede atacarlos.

Durante una partida sólamente se muestra la información de la simulación de un jugador. Dicha información incluye el atacante y el objetivo, su estrategia de ataque, así como el total jugadores activos. El usuario

elige cuál jugador visualizar al ejecutar el programa y cuál será la estrategia de ataque para ese jugador. Si ocurriera que el Mario al cual se eligió para ver su partida muere, se debe elegir otro Mario y finalmente, cuando sólamente queda 1 jugador activo, la batalla (programa) termina y dicho jugador es anunciado como el ganador.

3. Resultados esperados

Al terminar la simulación, el programa anuciará cual es el Mario ganador del Battle Royale y se podrá volver a correr la simulación para una nueva partida del Battle Royale.

Conclusiones

Este proyecto se centró en la implementación de un simulador de un Battle Royale de Procesos utilizando MPI, donde se una versión muy simplificada del juego Super Mario Bross 35. Para esto se implementó varios procesos simultáneamente simulando el juego e interactuando entre ellos, donde cada jugador virtual simuló el recorrido de Mario a lo largo del primer mundo (1-1) de su versión original.