Proyecto 1

Scheduling Cars

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
Principios de Sistemas Operativos
I Semestre 2025

Objetivo

Crear una biblioteca de hilos, con el fin de ejecutar distintas tareas de manera simultánea y con ello calendarizar las según se requiera.

Atributos a evaluar

- Trabajo individual y en equipo. Se requiere que funcione de manera efectiva como individuo y como miembro o líder en equipos diversos e inclusivos y en entornos multidisciplinarios, cara a cara, remotos y distribuidos.

Motivación

En el contexto de los sistemas operativos, los diferentes paralelismos que existen son campos de acción de un ingeniero en computadores, es por esto que se debe de comprender los conceptos básicos relacionados a estos temas. Los procesos son el factor común de todos los tipos de paralelismo, por lo que la creación y manipulación de estos debe ser impecable por parte de un ingeniero en computadores. Este proyecto busca implementar los hilos en espacio de usuario de tal manera que sea lo más eficiente posible de acuerdo con la funcionalidad solicitada. La abstracción de todo el ambiente que se debe de tener para controlar cualquier evento se vuelve esencial para un correcto funcionamiento del programa.

Descripción del problema

CEThreads

Se deberá **reimplementar** la biblioteca Pthreads (utilizando las directivas de C, se aclara que un wrapper o interface no es reimplementación) con el nombre CEthreads, la cual incluye. al menos, las siguientes funciones:

- ***CEthread_create***
- ***CEthread_join***
- ***CEmutex_init***
- ***CEmutex_destroy***
- ***CEmutex_unlock***

Calendarizadores

Los calendarizadores se encargan de ordenar la cola de “listo”, para que el hilo que se debe de ejecutar siempre esté de primero y los demás ordenados. El algoritmo de calendarización deberá ser un parámetro configurable por el usuario final.
Los algoritmos a implementar son:

- RR.
- Prioridad. Se necesitará un parámetro extra para indicar la prioridad.
- SJF. Necesitará un parámetro extra para indicar el tiempo de cada carro
- FCFS
- Tiempo real. Necesitaría el tiempo máximo que debe durar en pasar la calle.

Ejemplo de uso de la biblioteca CE Threads

Con el fin de visualizar el funcionamiento de la biblioteca CE Threads, se implementará una interfaz gráfica en la cual habrá varios vehículos que intentarán cruzar una zona crítica. La carretera es el medio por donde transitan los carros, para ir de un lado a otro. Es de ambos sentidos, pero solo puede usarse en un sentido a la vez, de acuerdo con el algoritmo que se especifica más adelante.

En cada extremo de la carretera se formará una fila de carros que esperan su turno para transitar por ella. Esta fila debe estar ordenada de acuerdo con el calendarizador elegido por el usuario. La cantidad de carros a representar en cada lado de la carretera debe ser especificada por el usuario.

La carretera tiene una longitud definida por el usuario, así como también la velocidad de los carros. La Figura 1 muestra un ejemplo de lo que se espera tanto a nivel de hardware como de interfaz.

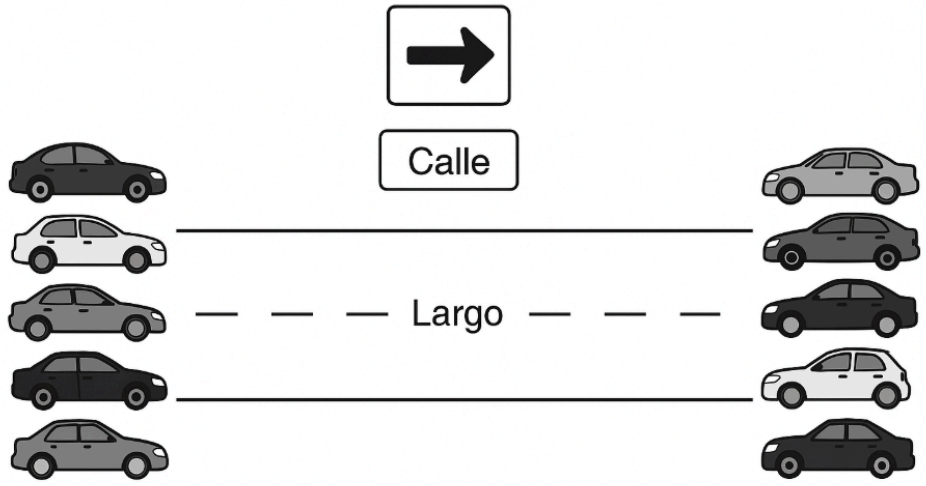

Figura 1: Ejemplo de Interfaz

El flujo de los carros se controlará con alguno de los siguientes algoritmos de flujo:

- Equidad: consiste en establecer un parámetro W (indicado por el usuario) que indica cuántos carros deben de pasar de cada lado. Es decir, se inicia permitiendo el paso de W carros de izquierda a derecha, y luego W carros de derecha a izquierda. En caso de que en alguno de los lados no haya carros, se debe garantizar el flujo de vehículos desde el lado donde sí los haya.
- Letrero: consiste en un letrero (izquierda - derecha) que indica el sentido habilitado de la vía. Cuando el letrero está en “izquierda”, pasan los carros del lado izquierdo; si está en “derecha”, pasan los del lado derecho. El letrero cambia automáticamente cada cierto tiempo, el cual es definido por el usuario al iniciar el programa.
- FIFO: no hay control explícito del flujo, sin embargo, cruzará el vehículo que llegó de primero, sin importar si llegó a la izquierda o a la derecha.

En todos los casos, los carros no pueden pasar mientras haya otros circulando en sentido contrario, con el fin de evitar accidentes. Cabe destacar que por ningún motivo puede haber colisiones.

Se debe crear un archivo de configuración para la calle, en el cual se especifique lo siguiente:

- Método de control de flujo.
- El largo de la calle.
- La velocidad de los carros.
- Cantidad de carros ordenados en la cola de listos.
- El tiempo que el letrero cambie (en caso de que aplique).
- El parámetro W (en caso de que aplique).
- Cualquier otro que sea necesario pasar como parámetro.

Sobre los vehículos

Los carros se generarán desde los extremos de la carretera (por ejemplo, desde la izquierda o desde la derecha), y la función de todos es atravesar la carretera para llegar al otro lado. Cada carro debe ser representado como un hilo, por lo que se generará un nuevo hilo cada vez que se cree un carro. Debe existir algún mecanismo sencillo para generar nuevos carros (puede ser usando un par de teclas del teclado) y se debe indicar desde cuál extremo (izquierda o derecha) se desea generalo.
Los carros se generan con velocidades ligeramente distintas de acuerdo con su tipo. Los carros pueden ser de tres tipos:

- Normales: sin tratamiento especial. Son los más lentos.
- Deportivos: más rápidos que los normales.
- Emergencia: tienen tratamiento prioritario, ya que representan vehículos de urgencia (como patrullas, ambulancias o bomberos). Son los más rápidos y deben ser tratados como un sistema de tiempo real hard, es decir, no pueden esperar más allá de cierto tiempo para cruzar.

La generación de carros puede hacerse de dos formas:

- Con carga definida: el usuario introduce antes de ejecutar el programa un conjunto de carros que se deben generar en cada extremo de la carretera, indicando el tipo de carro.
- Por medio de teclado: se inicia la ejecución del programa, y en cualquier momento se puede generar un nuevo carro de cualquier tipo desde cualquiera de los dos extremos de la carretera.

Interfaz gráfica

Se deberá crear una interfaz gráfica que muestre lo descrito en las secciones anteriores. Se espera una pantalla similar a la figura 1. Cabe destacar que la interacción entre el programa y el usuario debe ser lo más sencilla posible. Además, se evaluará la fluidez de las animaciones, así como el cumplimiento de las restricciones de movimiento. Cabe destacar que por ninguna circunstancia los carros pueden chocar o rebasar a otros. Esto implica que dos carros no pueden estar en el mismo punto al mismo tiempo.

Todos los elementos (carros, calle, señales/letrero, extremos de entrada/salida, etc.) deben estar representados gráficamente en pantalla. El usuario podrá cancelar el programa presionando la tecla w, y este debe cerrarse de manera elegante, asegurando que todos los hilos terminen correctamente y que los recursos utilizados se liberen adecuadamente.

Analogías con los recursos computacionales

- La calle juega el papel del CPU.
- Los carros son diferentes procesos (cada uno es un hilo).
- Las posiciones son recursos computacionales.
- Los calendarizadores son los algoritmos que se utilizan en un SO.
- Los algoritmos de flujo son análogos a las políticas de un SO.

Requerimientos técnicos

- La interfaz gráfica puede hacerse en cualquier lenguaje, el cual solo posea un **único hilo**.
- Este proyecto se debe realizar en el lenguaje de programación C. Pueden utilizar las bibliotecas que sean necesarias a excepción de la de hilos.
- Debe ser implementado en Linux y se debe proporcionar un Makefile, que genere e instale lo necesario.
- No se permiten soluciones “alambradas”.
- Se debe prestar especial atención a los errores de acceso a memoria o utilización de recursos. Es inaceptable el error ***segmentation fault*** o ***Core dumped*** se penaliza con 5 puntos cada error de este tipo.

Documentación externa

La documentación externa será un documento en formato PDF, estilo IEEE-Trans de máximo 5 páginas. El documento debe tener, por lo menos, los siguientes capítulos.

- Abstract: Síntesis del proyecto, generalmente contiene entre 150-200 palabras.Debe indicar en qué consiste el proyecto o tarea, como se realizó el proyecto o tarea y el principal hallazgo o conclusión.
- Introducción: Teoría necesaria, breve descripción del proyecto y qué es lo que se espera en el escrito.
- Ambiente de desarrollo: Se debe especificar todo lo que se ocupa para ejecutar el proyecto.
- Atributos: En esta sección deben de describirse cuales atributos fueron reforzados durante el desarrollo del proyecto. Para el atributo de Trabajo individual y en equipo se debe especificar 7 puntos (Se debe colocar pregunta y respuesta), los cuales son los siguientes:
    - Indicar las estrategias para el trabajo individual y en equipo de forma equitativa e inclusiva en las etapas del proyecto (planificación, ejecución y evaluación).
    - Indicar la planificación del trabajo mediante la identificación de roles, metas y reglas.

        - Indicar cuáles acciones promueven la colaboración entre los miembros del equipo durante el desarrollo del proyecto.
        - Indicar cómo se ejecutan las estrategias planificadas para el logro de los objetivos.
        - Indicar la evaluación para la el desempeño del trabajo individual y en equipo
        - Indicar la evaluación para las estrategias utilizadas de equidad e inclusión.
        - Indicar la evaluación para las acciones de colaboración entre los miembros del equipo
- Diseño: Diagramas UML, secuencia, arquitectura, imágenes, descripciones entre otros, todo lo que sea necesario para entender de una mejor manera el diseño y funcionamiento del proyecto. Se espera como mínimo el de arquitectura, componentes, UML y secuencia.
- Instrucciones de cómo se utiliza el proyecto.
- Conclusiones: se espera la generalización de los resultados, recordar que las conclusiones no son un resumen de lo que se realizó.
- Sugerencias y recomendaciones.
- Referencias

Entregables

- Código fuente con documentación interna.
- Documentación.
- Archivos necesarios para ejecutar el programa

Aspectos administrativos

- Para la revisión del proyecto se debe entregar tanto la documentación como la implementación del software.
- No se reciben trabajos después de la hora indicada.
- En la revisión del proyecto pueden estar presentes el coordinador y asistente.
- Es responsabilidad del estudiante proveer los medios para poder revisar la funcionalidad del software, por ejemplo, si no se realiza la interfaz, se debe

de proporcionar otro medio para la verificación, de lo contrario la nota será cero en los rubros correspondientes a la funcionalidad faltante.

- En caso de que alguna parte de la especificación sea confusa o ambigua, el grupo de trabajo puede agendar una cita con el profesor para aclarar cada aspecto.
- Fecha de asignación: 9 de abril del 2025
- Fecha de entrega: 7 de mayo del 2025
- Grupos: 3 o 4 personas

Rúbrica de evaluación

Funcionalidad (80%)	Valor (%)
Biblioteca de hilos Se implementa la biblioteca de hilos de manera correcta con todas las funcionalidades solicitadas.	20
Calendarización Los 5 algoritmos funcionan de manera correcta y se pueden probar de alguna manera	15
Integración Todas las funcionalidades están integradas de manera correcta	10
Usuario y archivos El sistema se puede configurar en modo manual o en modo de archivos	5
Sincronización y robustez Todos los hilos se crean y se mueven de manera correcta y sin ocurrir choques. Los algoritmos de flujo funcionan de manera correcta	15
Interfaz gráfica Se representan letreros, movimientos, vehículos, calle, así como la cola de listos.	15
Documentación externa (20%)	
Abstract	1
Introducción	1
Ambiente de desarrollo	2
Atributos	5

Diseño	7
Instrucciones	1
Conclusiones	1
Sugerencias y recomendaciones	1
Referencias	1