Trabajo Práctico Grupal de Algoritmos y Programación II

Primer Cuatrimestre 2025

Objetivo

El objetivo de este trabajo es aplicar los conceptos aprendidos en clase en la resolución de un problema práctico, demostrando la capacidad de diseñar e implementar soluciones eficientes utilizando estructuras de datos y algoritmos avanzados.

Enunciado

Desarrollar un **Sistema Integrado de Gestión de Operaciones Aeroportuarias** (SIGOA) que abarque la gestión de las operaciones de un aeropuerto, utilizando diversas estructuras de datos y técnicas algorítmicas para optimizar los procesos de check-in, gestión de carga y control de vuelos.

Objetivos Específicos

- Diseñar e implementar un sistema que simule las operaciones de un aeropuerto, incluyendo el check-in de pasajeros, la gestión de carga y el despacho de vuelos.
- Utilizar estructuras de datos adecuadas (pilas, colas, listas, colas de prioridad, árboles binarios) para representar la información y gestionar los procesos.
- Aplicar algoritmos de búsqueda, ordenamiento y optimización para mejorar la eficiencia del sistema.
- Implementar técnicas de compresión de datos para el almacenamiento eficiente de la información del vuelo.
- Evaluar el rendimiento del sistema mediante pruebas con diferentes escenarios de tráfico aéreo y volumen de pasajeros/carga.
- Documentar el diseño, la implementación y los resultados del sistema en un informe técnico detallado.

Descripción del Sistema

1. Check-in de pasajeros

- Cada vuelo tiene un número de vuelo, una fecha y hora de salida programada, y una lista de pasajeros asignados y abre el check-in 2 horas antes de la salida del vuelo.
- Cada pasajero se identifica con su número de documento, que puede ser DNI o pasaporte. El sistema debe permitir buscar un pasajero y recuperar su número de reserva, nombre, apellido, categoría de pasajero (platino, oro o plata) y el número de vuelo al que está asignado. En una reserva pueden haber hasta 5 pasajeros.
- Los pasajeros llegan al aeropuerto y se dirigen a los mostradores de check-in. (Se debe simular la llegada de los pasajeros al aeropuerto y su espera en la cola de check-in). Se estima que cada pasajero tarda entre 1 y 5 minutos en realizar el check-in.

- Se implementa una única cola para gestionar la espera de los pasajeros en el mostrador de check-in y se pueden habilitar varios mostradores para atender a los pasajeros simultáneamente. La cantidad de mostradores habilitados puede variar durante la simulación.
- Los pasajeros pueden despachar su equipaje y recibir su tarjeta de embarque, siempre y cuando se encuentren en la lista de pasajeros del vuelo.
- El sistema debe emitir un ticket de equipaje por cada bulto despachado, que contenga el número de vuelo, el número de documento del pasajero, el número de bultos despachados y el peso total del equipaje.
- El vuelo puede tener pasajeros en lista de espera. Cuando la aerolínea cierra el proceso de check-in (por ejemplo, 45 minutos antes de la salida), si hay asientos disponibles, se asignan a los pasajeros en lista de espera. La asignación se realiza por prioridad (platino > oro > plata > no frecuente) y, dentro de cada prioridad, por orden de llegada al check-in.

2. Gestión de cargas

- Los aviones pueden transportar carga, la cual debe ser gestionada de manera eficiente.
- Los paquetes de carga se encuentran en el aeropuerto, identificados por el destino, el peso y el volumen.
- Cada aeronave tiene una capacidad máxima de carga en peso y volumen, y se deben respetar estas restricciones al cargar los paquetes.
- Los paquetes se deben asignar a la aeronave tratando de maximizar el uso del espacio disponible y respetando las restricciones de peso. Se deberá implementar un algoritmo que intente optimizar la carga (por ejemplo, utilizando alguna heurística de empaquetamiento).
- La carga se realiza una vez finalizado el check-in y hasta 1 hora antes de la salida del vuelo. Una vez finalizada la carga, se procede al embarque de los pasajeros.

3. Embarque de pasajeros

- Los pasajeros deben embarcar en el avión de acuerdo a un orden establecido.
- Los pasajeros de la categoría platino, oro y plata tienen embarque prioritario y pueden abordar en cualquier momento una vez que se inicia el proceso de embarque.
- Los pasajeros restantes embarcan por zonas, primero se habilita el embarque de la zona 1, luego la zona 2 y así sucesivamente.
- Cada vuelo se divide en al menos dos zonas que se indican en la tarjeta de embarque. La asignación de zonas se basa en la ubicación del asiento dentro de la aeronave, según la configuración definida para cada modelo de avión. Los que tienen asientos asignados en la parte de atrás corresponden a la zona 1, los del medio a la zona 2 y los de adelante a la zona 3. La asignación de zonas debe realizarse al momento del check-in, utilizando la información del archivo `configuracion_asientos.txt`.
- Una vez que todos los pasajeros han embarcado o se ha alcanzado la hora límite de embarque (por ejemplo, 15 minutos antes de la salida), se cierra la puerta del avión y el avión queda en estado listo para despegar, esperando las instrucciones del control de tráfico aéreo.
- Llegada la hora límite de embarque, si algún pasajero no se presenta, se lo da de baja de la nómina de pasajeros.
- Toda la información relativa al vuelo se registra en un archivo de texto al finalizar el embarque. Esta información debe incluir el número de vuelo, la fecha y hora de salida programada, la fecha y hora de partida real, la lista de los pasajeros embarcados (con su número de documento y categoría), la lista de bultos despachados en calidad de equipaje (asociados a cada pasajero), la lista de los pasajeros que no se presentaron, la lista de los pasajeros que quedaron en lista de espera (indicando si finalmente embarcaron o no), y la lista de los paquetes de carga embarcados (con su destino, peso y volumen). El archivo se comprime utilizando el

algoritmo de codificación Huffman y se almacena en el servidor del aeropuerto. Este archivo comprimido será uno de los **archivos de salida** del sistema.

4. Despacho de vuelos

- Los vuelos deben ser despachados de acuerdo a un horario programado.
- Se implementa una cola de prioridad para gestionar el despacho de vuelos, considerando la hora de salida programada (mayor prioridad para las horas más tempranas) y el estado de preparación del vuelo (mayor prioridad para los vuelos listos para despegar).
- Antes de despachar un vuelo, se debe calcular la línea del horizonte para garantizar la seguridad del despegue. La línea del horizonte se calcula a partir de una lista de edificios y obstáculos en el área del aeropuerto. La entrada para calcular la línea del horizonte es un archivo de texto (`edificios.txt`) que contiene la posición inicial (x1), altura (h) y posición final (x2) de cada edificio u obstáculo. La salida debe ser un archivo de texto que contenga una serie de puntos (posición x, altura y) que representen la línea del horizonte. Este archivo con la línea del horizonte será otro de los **archivos de salida** del sistema.

El sistema debe simular la operación completa de un aeropuerto, desde la llegada de los pasajeros y la carga hasta el despacho de los vuelos. Los datos iniciales para la simulación se deben leer desde los archivos de texto proporcionados en la carpeta `data`. La simulación debe avanzar en el tiempo, gestionando los eventos (llegada de pasajeros, inicio de check-in, fin de check-in, inicio de carga, fin de carga, inicio de embarque, fin de embarque, despacho de vuelos) de manera lógica.

En resumen, la aplicación deberá generar los siguientes archivos de salida:

- **Archivo de información del vuelo comprimido:** Un archivo `.huff` (o la extensión que corresponda a la implementación del algoritmo Huffman) que contenga la información detallada del vuelo una vez finalizado el embarque.
- **Archivo de la línea del horizonte:** Un archivo de texto (`linea_horizonte.txt` o un nombre similar) que contenga los puntos (posición x, altura y) que definen la línea del horizonte calculada.

Requerimientos Adicionales

- El sistema debe simular la operación completa de un aeropuerto, utilizando datos de archivos de texto definidos por el grupo. La simulación debe ser controlada por eventos y debe mostrar en la consola los eventos principales que ocurren en el sistema (por ejemplo, “Pasajero X llega al check-in”, “Vuelo Y comienza el check-in”, “Vuelo Z es despachado”).
- El sistema debe ser implementado en lenguaje de programación Go.
- El proyecto debe ser desarrollado utilizando buenas prácticas de programación, incluyendo la correcta modularización del código, el uso de estructuras de datos abstractas (interfaces) cuando sea apropiado y la documentación interna (comentarios en el código).
- El informe técnico debe incluir:
  - Diagramas de las estructuras de datos utilizadas para representar las entidades del sistema (vuelos, pasajeros, carga, etc.) y para gestionar los procesos (colas, listas, etc.).
  - Descripción detallada de los algoritmos implementados para las tareas clave (check-in, asignación de carga, cálculo de la línea del horizonte, compresión Huffman, asignación de zonas de embarque, etc.).
  - Análisis de la complejidad temporal y espacial de los algoritmos implementados.

- Resultados de las pruebas realizadas con diferentes escenarios de tráfico aéreo (bajo, medio, alto) y volumen de pasajeros/carga. Se deben incluir métricas relevantes para evaluar el rendimiento del sistema (por ejemplo, tiempo promedio de espera en el check-in, porcentaje de utilización de la capacidad de carga, tiempo total de procesamiento de un vuelo).
- Conclusiones sobre la eficiencia del sistema, identificación de posibles cuellos de botella y propuestas de mejoras.

Datos de Entrada

En la carpeta `data` se encuentran los archivos de texto que contienen ejemplos de los archivos de datos de entrada:

- `vuelos.txt`: Contiene la información de los vuelos programados, incluyendo número de vuelo, fecha y hora de salida programada (en formato AAAA-MM-DD HH:MM), destino (código IATA) y código de la aeronave asignada.
- `clientes.txt`: Contiene la información de los clientes de una aerolínea, potenciales pasajeros, incluyendo Nombre, Apellido, Número de Documento (DNI o pasaporte) y categoría (platino, oro, plata o ““).
- `aeropuertos.txt`: Contiene la información de los aeropuertos, incluyendo Provincia, Ciudad, Nombre del Aeropuerto y Código IATA (Código Internacional de 3 letras).
- `carga.txt`: Contiene la información de los paquetes de carga, incluyendo el destino (código IATA), peso (en kg) y volumen (en m^3).
- `reservas.txt`: Contiene la información de las reservas de los pasajeros, incluyendo el número de reserva, Número de Documento del pasajero, número de vuelo y estado de la reserva (confirmada, lista de espera).
- `edificios.txt`: Contiene la información de los edificios y obstáculos en el área del aeropuerto, incluyendo la posición inicial (x1 en metros), altura (h en metros) y posición final (x2 en metros) de cada edificio u obstáculo.
- `aeronaves.txt`: Contiene la información de las aeronaves, incluyendo el código de la aeronave, el modelo, la capacidad máxima de carga en peso (en kg) y la capacidad máxima de carga en volumen (en m^3), y el número total de asientos.
- `configuracion_asientos.txt`: Contiene la configuración de los asientos por zona para cada modelo de aeronave, incluyendo el código de la aeronave, la zona de embarque y el rango de números de asiento correspondientes (ej: `B738, 1, 1, 20`).

Como parte del trabajo práctico, se espera que los grupos generen sus propios archivos de entrada para realizar pruebas adicionales y simular diferentes escenarios. Se espera que los grupos generen al menos 3 escenarios diferentes (bajo, medio y alto tráfico aéreo) y que cada escenario contenga al menos 10 vuelos programados, 100 pasajeros y 50 paquetes de carga. Al menos 20 edificios para calcular la línea del horizonte y al menos 5 aeronaves diferentes.

Los archivos de salida generados por el sistema deben ser almacenados en la carpeta `output` y deben seguir un formato claro y consistente para facilitar su análisis posterior. Los archivos comprimidos generados por el algoritmo de Huffman se deben poder descomprimir y analizar para verificar la correcta compresión de los datos, para lo cual se deberá incluir una herramienta de descompresión de Huffman en el proyecto.

Entregables

1. **Código Fuente:**
    - Todo el código fuente de los programas, organizado en carpetas según la funcionalidad.

2. **Documentación:**
    - Documentación del código y las decisiones de diseño.
    - Un informe detallado con la explicación de la solución y los resultados obtenidos.
3. **Datos:**
    - Archivos generados durante la ejecución de los programas.
    - Ejemplos de entradas y salidas utilizadas para las pruebas.
4. **Presentación Oral:**
    - Una presentación oral explicando las soluciones implementadas y respondiendo preguntas sobre el código y los algoritmos utilizados. Las fechas de las presentaciones serán anunciadas posteriormente.

Fecha Límite de Entrega:

- Semana del 23 de junio. Luego del segundo parcial se realizarán las presentaciones orales.

Evaluación:

- La correcta implementación de los algoritmos y estructuras de datos.
- La eficiencia y claridad del código.
- La documentación y presentación del trabajo.
- La colaboración y el trabajo en equipo.

Instrucciones Adicionales para el Trabajo en Grupo

- **Formación de Grupos:** Cada grupo debe estar conformado por 4 estudiantes.
- **Distribución de Tareas:** Es importante que cada integrante del grupo tenga una responsabilidad clara. Se recomienda asignar tareas específicas para asegurar una distribución equitativa del trabajo.
- **Coordinación y Comunicación:** Utilizar herramientas de gestión de proyectos (como Trello o Asana) y comunicación (Slack o WhatsApp) para coordinar el trabajo y mantener una comunicación fluida.
- **Control de Versiones:** Utilizar Git para colaborar en el desarrollo del código y mantener un historial de cambios

> [!CAUTION] 🚨 **TODOS LOS INTEGRANTES DEL EQUIPO DEBERAN HACER COMMITS AL REPOSITORIO PARA DEMOSTRAR SU PARTICIPACIÓN EN EL DESARROLLO** 🚨