Proyecto Final

- **Asignatura:** Diseño y Análisis de Algoritmos
- **Año Académico:** 4to Año, Ciencia de la Computación

1. Introducción y Objetivos

El objetivo fundamental de este proyecto es aplicar de manera integral los conocimientos teóricos y prácticos adquiridos en la asignatura. Los estudiantes se enfrentarán a un problema computacionalmente difícil (del tipo NP-duro o NP-completo), llevando a cabo un ciclo completo de análisis que abarca desde su formalización matemática hasta el diseño, implementación y evaluación empírica de algoritmos de solución.

Este proyecto busca desarrollar habilidades críticas en:

- Modelado y formalización de problemas.
- Análisis riguroso de la complejidad computacional.
- Diseño de algoritmos para problemas complejos (aproximación, heurísticas).
- Implementación de software y evaluación experimental de su rendimiento.
- Comunicación técnica a través de informes y presentaciones.

2. Metodología y Formación de Equipos

- El proyecto se realizará en equipos de **dos (2) o tres (3) estudiantes** como máximo.
- A cada equipo se le asignará la descripción de un problema del mundo real, definido de manera **informal**.
- Opcionalmente, los equipos podrán proponer un problema de su propio interés. Dicha propuesta será analizada por el colectivo de la asignatura y, en caso de ser aprobada por su complejidad y alcance, podrá ser utilizada para el proyecto.

3. Fases y Tareas del Proyecto

El trabajo del equipo consistirá en un análisis exhaustivo del problema asignado, cubriendo las siguientes fases:

Fase 1: Formalización del Problema

Partiendo de la descripción informal, el equipo deberá:

- Traducir el problema a un modelo matemático preciso y sin ambigüedades.

- Definir formalmente las estructuras de datos de entrada, las restricciones que deben cumplirse y las propiedades de la salida deseada (solución óptima).

Fase 2: Análisis de Complejidad Computacional

El equipo deberá:

- Demostrar formalmente que el problema pertenece a la clase **NP-duro** o **NP-completo**.
- Esta demostración debe incluir una **reducción en tiempo polinomial** desde un problema canónico cuya complejidad sea conocida (ej. 3-SAT, Vertex Cover, Hamiltonian Cycle, TSP, etc.).

Fase 3: Diseño de Soluciones Algorítmicas

Dado que el problema es computacionalmente intratable, se deben explorar soluciones no exactas. El equipo deberá:

- **Diseñar e implementar un algoritmo de fuerza bruta** que encuentre la solución óptima. Este servirá como línea base para comparar resultados en instancias pequeñas.
- **Diseñar y proponer una solución eficiente** utilizando una o más de las siguientes técnicas:
  - **Algoritmos de Aproximación:** Con una garantía de rendimiento teórica demostrable.
  - **Heurísticas y Metaheurísticas:** Como algoritmos voraces (greedy), búsqueda local, etc.
  - **Kernelización** o **Esquemas de Aproximación en Tiempo Polinomial (PTAS)**, si se considera aplicable.

Fase 4: Implementación y Análisis Experimental

El equipo deberá:

- Implementar los algoritmos (fuerza bruta y la solución eficiente) en un lenguaje de programación a elección (se sugieren Python, C++, C#, Haskell, etc.).
- Diseñar y generar un conjunto robusto de **instancias de prueba**. Esto incluye la creación de un generador de casos aleatorios y la definición de casos específicos que pongan a prueba las aristas del problema.
- Realizar un análisis empírico comparando:
  - La **calidad de la solución** obtenida por el algoritmo eficiente frente a la solución óptima de la fuerza bruta.
  - Los **tiempos de ejecución** de ambos enfoques.
  - Identificar el **tamaño máximo de instancia** que el algoritmo de fuerza bruta puede resolver en un tiempo razonable.

- Analizar el comportamiento de la solución propuesta: ¿en qué tipo de instancias funciona mejor o peor? Si se usó una heurística, ¿cuándo encuentra la solución óptima y cuándo no?

4. Entregables

1. **Informe Formal (en LaTeX):** Un documento técnico, bien estructurado y redactado, que detalle exhaustivamente cada una de las fases del proyecto. La calidad, claridad y rigor del informe son cruciales para la evaluación.
2. **Repositorio en GitHub:** Un repositorio de código que contenga:
   - El código fuente completo de todas las implementaciones, debidamente documentado y comentado.
   - Un archivo `README.md` con instrucciones claras para compilar y ejecutar los algoritmos.
   - El generador de instancias y los casos de prueba utilizados en el análisis experimental.
   - (Opcional pero muy recomendado) Cualquier script o herramienta utilizada para la visualización de los problemas o sus soluciones.

5. Evaluación

La nota final del proyecto será el resultado de la **presentación oral**. Si bien el informe y el repositorio son entregables obligatorios que servirán como soporte fundamental, la calificación definitiva se determinará en la defensa del trabajo.

Durante la presentación, además de exponer su solución, la evaluación podrá incluir **preguntas adicionales sobre cualquier contenido de la asignatura**, cuya naturaleza podrá depender de los resultados y el desempeño previo de los estudiantes.

Aunque el proyecto se centra en la evaluación de los contenidos de **complejidad computacional y NP-completitud**, también servirá para evaluar la aplicación de cualquier otro tema del curso que se haya incluido en la solución. Esto brinda la oportunidad de demostrar maestría en áreas como: **programación dinámica, algoritmos voraces (greedy), reducciones polinomiales, análisis de cotas inferiores, algoritmos de flujo en redes**, etc.

6. Fechas Importantes

- **Asignación de problemas:** La fecha máxima para la selección o asignación de un problema es el **viernes 31 de octubre**.
- **Fecha Límite de Entrega (Informe y Repositorio):** La entrega final se realizará durante la **última semana del curso**.

- **Presentaciones Orales:** Las exposiciones se llevarán a cabo durante el **período de exámenes finales**. Aquellos equipos que finalicen su proyecto con antelación y deseen presentarlo, podrán coordinar una fecha con los profesores.

ALEJANDRO PIAD MORFFIS

Signed by: ALEJANDRO PIAD MORFFIS

Date: 2025-10-21 13:18:29 CDT