REGLAS DEL PROYECTO LONGITUDINAL CURSO DE ALGORITMOS AVANZADOS

Mauricio Velasco

1. Entregables del Proyecto Longitudinal.

Cada proyecto consiste en la elaboración de dos entregables: Un *demo interactivo* y un documento de *sustento teórico* acerca alguno de los algoritmos del curso.
El *demo interactivo* debe:

1. Permitir que el usuario ingrese los datos de entrada del algoritmo (de forma cómoda, ya sea mediante una interfaz desarrollada por uds. con este fin ó mediante el uso de hojas tipo json/excel).

2. Ejecutar el algoritmo de manera correcta sobre los datos de entrada dados.

3. Visualizar el resultado del algoritmo.

4. Mostrar una explicación de los pasos principales del mismo.

5. Ser codificado en su totalidad por los estudiantes del curso en un repositorio de github (puede usar cualquier fuente adicional pero la totalidad del código debe estar en el repositorio).

Como referente de lo que se busca ver las páginas https://www.redblobgames.com/articles/visibility/ (mover el círculo amarillo que aparece al final) y https://ncase.me/sight-and-light/.
Los ejemplos anteriores fueron hechos con vue.js (ver detalles en https://www.redblobgames.com/making-of/circle-drawing/) pero para su implementación pueden usar la infraestructura que quieran. Una opción sencilla es Jupyter Notebooks (ver https://hex.tech/blog/beginners-guide-to-python-notebooks/ y https://www.dataquest.io/blog/jupyter-notebook-tutorial/).

Por otro lado el *Documento de sustento teórico* del proyecto debe:

1. Describir el problema de manera matemáticamente precisa.
2. Describir el algoritmo utilizado de manera matemáticamente precisa.
3. Contener una demostración de la corrección del algoritmo utilizado ó una demostración de la validez de alguna cota de complejidad para el algoritmo.
4. Contener una discusión de aplicaciones potenciales del algoritmo a problemas de interés real (idealmente en el contexto nacional Uruguayo ó regional).
5. Tener una presentación profesional (y en particular debe ser escrito en LaTex).

Como inspiración acerca de la manera en la que el documento debe ser escrito, el formato, profundidad técnica, etc. ver la sección *What is...?* de la revista Notices of the American Mathematical Society. Por ejemplo https://finmath.stanford.edu/~cgates/PERSI/papers/what-is.pdf o cualquier otro de la lista http://arminstraub.com/math/what-is-column que sea de su interés.

2. Reglas y criterios de evaluación.

El proyecto es un trabajo en grupo longitudinal a ser realizado durante todo el semestre. Debe cumplir las siguientes reglas:

1. Todo grupo debe ser de 3 ó 4 estudiantes.
2. El algoritmo a tratar durante el proyecto es escogido por los estudiantes (ver algunos temas posibles sugeridos en la sección siguiente).
3. La evaluación del proyecto consistirá de tres entregas (y la nota de cada entrega será la misma para todos los integrantes del grupo). Las entregas ocurrirán en las fechas asignadas en la página web del curso y estarán distribuidas así:
    a) Entrega 1 (Inicios de semestre): Cada grupo debe entregar un documento con los nombres de los integrantes, el tema escogido y al menos dos referencias bibliográficas que hayan consultado sobre el tema.

b) Entrega 2: (Mediados de semestre) Cada grupo hace una presentación oral de 20 mins (con slides) mostrando a sus compañeros el estado actual de su desarrollo en el proyecto (tanto del demo como del documento teórico), enfatizando los logros ya obtenidos y las dificultades encontradas durante el desarrollo. La asistencia a las presentaciones de sus compañeros es obligatoria pues esperamos crear un ambiente que permita dar feedback constructivo a otros grupos.

c) Entrega 3: (Final del semestre) Cada grupo hace una presentación oral de 20 mins mostrando a sus compañeros versiones finales del proyecto (tanto del demo como del documento teórico). Adicionalmente cada grupo debe entregar al instructor el documento teórico (en .pdf enviado por correo) y el demo (mediante el link de github).

3. Posibles proyectos sugeridos

En esta sección escribo algunas direcciones sugeridas en las que podrían dirigir su proyecto.

1. Topological sorting de DAGs y sus aplicaciones https://en.wikipedia.org/wiki/Topological_sorting.

2. Escriba un programa que proponga y resuelva Sudokus ó algunas de sus variantes https://en.wikipedia.org/wiki/Sudoku.

3. Resolviendo laberintos con Dijkstra vs con A^* (comparar los dos) https://towardsdatascience.com/solving-mazes-with-python-f7a412f2493f y http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html.

4. El modelo de asignación de escuelas públicas en la ciudad de Nueva York se basa en encontrar un *stable matching*. Cómo funciona este algoritmo y qué aplicaciones de matching se les ocurren en el Uruguay? Ver https://icerm.brown.edu/video_archive/?play=3027 NOTA: Este es un tema de investigación activo muy interesante. El link es una clase que Yuri Faenza dió en Enero 2023 en ICERM. Ver tambien https://project.dke.maastrichtuniversity.nl/easss/wp-content/uploads/2018/07/EASSS_Tutorial.Stable_Matchings.Web_.pdf

5. Cuál es el camino más corto para recorrer todas las ciudades del Uruguay? Usar branch-and-bound para resolver el TSP en el mapa de Uruguay actualizando https://www.math.uwaterloo.ca/tsp/world/countries.

html, por ejemplo icorporando los tiempos de googleMaps. Ver tambien https://www.math.uwaterloo.ca/tsp/data/art/ para otras posibilidades.

6. Analizar alguna versión del algoritmo de Viterbi y alguna de sus aplicaciones https://en.wikipedia.org/wiki/Viterbi_algorithm

7. Algoritmos genéticos para resolver el Vehicle Routing Problem (VRP) https://pubsonline.informs.org/doi/10.1287/opre.1120.1048. Qué aplicaciones podrían tener los VRPs en Uruguay?

8. Construya un algoritmo que resuelva un cubo de Rubik de manera óptima formulándolo como un problema de distancia mínima en grafos. Ver https://web.mit.edu/sp.268/www/rubik.pdf.

NOTA: Las de arriba son sólo algunas sugerencias de direcciones posibles. Siéntanse en la libertad de dirigir el proyecto hacia cualquier algoritmo de su interés relacionado con el curso (esté o no en la lista de arriba).