PROGRAMA DE ASIGNATURA	
ASIGNATURA: Matemática Discreta II	**AÑO:** 2023
CARACTER: Obligatoria	**UBICACIÓN EN LA CARRERA:** 3° año 1° cuatrimestre
CARRERA: Licenciatura en Ciencias de la Computación	
REGIMEN: Cuatrimestral	**CARGA HORARIA:** 120 horas

ASIGNATURA: Matemática Discreta II	**AÑO:** 2023
CARACTER: Obligatoria	**UBICACIÓN EN LA CARRERA:** 4° año 1° cuatrimestre
CARRERA: Licenciatura en Matemática Aplicada	
REGIMEN: Cuatrimestral	**CARGA HORARIA:** 120 Horas.

FUNDAMENTACIÓN Y OBJETIVOS

Esta materia aborda temas de Matemática Discreta, Teoria de Códigos de Corrección de Errores, Teoría de Complejidad mas rudimentos de inteligencia artificial.

La parte principal de la materia es el estudio de algoritmos sobre grafos y networks, y especialmente el análisis de la corrección y las complejidades de los mismos.

El objetivo de esta parte es que los estudiantes comprendan que en muchas aplicaciones no basta dar un algoritmo sino que hay que demostrar su correctitud, dar una cota de su complejidad y demostrarla.

Esto es una mezcla de algoritmia y matemática.

El proyecto de programación se basa en esta parte para aprender las dificultades de traspasar elementos teóricos a codificaciones concretas.

Ademas de esta parte central la materia tambien cubre códigos de corrección de errores porque es un tema que necesitan en la materia Redes, cubre P-NP porque encaja bien con la primera parte viendo que a veces no hay algoritmos polinomiales para resolver problemas y Algoritmos Genéticos para ejemplificar un tema de inteligencia artificial y explicar un posible curso de acción en los casos en que no existen algoritmos polinomiales.

El total de unidades para el 2022 son 6:
Coloreo de Grafos
Fundamentos de inteligencia artificial
Flujos Maximales
Matchings
Códigos de corrección de errores.
P-NP

CONTENIDO

P-NP
La clases P y NP.
Ejemplos. El problema SAT.
El problema k-COLOR.

Reducci\ón polinomial.
Las clases de problemas NP-hard y NP-completo.

Teorema de Cook (sin prueba): SAT es NP-completo.

Teorema: 3-SAT es NP-completo.

Teorema: 3-COLOR es NP-completo.

Propiedades:(si el tiempo lo permite)
2-SAT esta en P. Horn-SAT esta en P.

Códigos de corrección de errores.
Códigos de corrección de errores.
Definiciones básicas.
Distancia de Hamming.
Detección y Correción de errores.
Ejemplos de códigos.
Chequeo de paridad.
Códigos de repetición.
Cota de Hamming.

Códigos Lineales.
Propiedad: Si C lineal entonces delta(C) es igual al mínimo peso no nulo.

Matrices Generadoras.
Códigos lineales como espacios filas de una matriz.

Códigos lineales como nucleos de matrices. Matrices de chequeo.

Equivalencia entre matrices generadoras y de chequeo. Propiedad: todo código lineal tiene una matriz de chequeo.

Proposición: Si en la matriz de chequeo no hay columnas repetidas ni nulas entonces el código correspondiente corrije al menos un error.
Generalización de esta propiedad a corrección de mas errores:
Teorema: Si H es una matriz de chequeo de C, entonces
delta(C)=Min{j: existe un conjunto de j columnas linealmente dependientes de H}

Algoritmo para correjir un error.
Códigos de Hamming.
Códigos perfectos.
Propiedad: Hamming es perfecto.
Singleton Bound.
Códigos MDS

Códigos Cíclicos.

Rotación de una palabra.
Códigos cíclicos.
Códigos ciclicos mirados como polinomios.

Propiedad:

todo código lineal binario tiene un único polinomio no nulo de menor grado.

Definición de Polinomio generador de un código cíclico.

Propiedades del polinomio generador.

Uso del polinomio generador para codificacion:
dos m\étodos.
Matrices generadoras asociadas a los dos m\étodos
Obtenci\ón en forma directa a partir del polinomio generador de una matriz de chequeo con la identidad a izquierda. Polinomio chequeador.

Corrección de errores: error trapping. (si el tiempo lo permite)

Códigos de ReedSolomon (si el tiempo lo permite)

Matchings
Matchings en grafos bipartitos,
Matchings perfectos y Matchings completos.
Ejemplos.
Algoritmo para encontrar matchings como aplicación de los algoritmos para encontrar flujos maximales. Modificaciones.
Uso de matrices.
Definicion de Gamma (S).
Condición de Hall.
Teorema de Hall.

Teorema del Matrimonio. (Todo grafo bipartito regular tiene un matching perfecto).

Problemas de Matchings Optimos en grafos bipartitos con pesos.

Resolucion del ``bottleneck problem'': problema
del asignamiento optimo cuando se desea minimizar el maximo (o maximizar el minimo) de los pesos.

Resolución del problema del asignamiento óptimo cuando se desea minimizar (o maximizar)
la suma de los pesos:
Algoritmo Húngaro.
Codificación de complejidad O(n al cubo) del algoritmo Húngaro.

Flujos Maximales
Grafos Dirigidos.
Ejemplos.
Networks.(redes)
Flujos sobre redes.
Valor de un flujo.
Flujos maximales.
Diversos ejemplos.
Algoritmo Greedy para encontrar flujo maximal.
Ejemplo donde no necesariamente encuentra flujo maximal.
Definición de corte y capacidad de un corte.
Caminos aumentantes de Ford-Fulkerson.
Algoritmo de Ford-Fulkerson.
Propiedad: Al aumentar el flujo a lo largo de un camino aumentante de Ford-Fulkerson lo que se

obtiene sigue siendo flujo.

Max Flow Min Cut Theorem:
a)El valor de todo flujo es menor o igual que la capacidad de todo corte.
b)Si f es un flujo, las siguientes afirmaciones son equivalentes:
1) f es maximal.
2) Existe un corte S tal que v(f)=cap(S).
3) No existen f-caminos aumentantes.

Ejemplos de aplicación
del algoritmo de Ford-Fulkerson.
Debilidades del algoritmo de Ford-Fulkerson:
Ejemplo donde la complejidad no depende del numero de vertices o lados.
Ejemplo donde el algoritmo no termina.
Refinamientos:
Algoritmos fuertemente polinomiales:
Algoritmo de Edmonds-Karp. Complejidad.
Algoritmo de Dinic o Dinitz. Complejidad de sus 2 versiones.
Algoritmos de pre-flow/push: algoritmo ``wave'' de Tarjan. Complejidad.

Fundamentos de inteligencia artificial
Algoritmos de Busqueda.
Hill Climbing.
Simulated Annealing.
Algoritmos Genéticos:
Codificación del problema.
Fitness.
Reproducción de Población.
Terminación.

Selecci\ón, Crossover, Mutaci\ón, Reemplazo.

Algunas posibilidades de Mutación.

Algunos posibilidades de Crossover.
Single point, double, multiple points o mascara.
Crossover en el caso de permutation based codifications: crossover básico, Partial Mixing Crossover y Ciclico.

Algunas posibilidades de Selecci\ón:
Ruleta, SUS, Rank-based selection
Sigma based selection.

Otras posibilidades de estructura: catástrofes e islas. con migraciones.

Coloreo de Grafos
Repaso de la noción de grafo.
Notaciones.
Coloreo de Grafos.
Numero cromático.
Algoritmo de fuerza bruta.
Problema k-Color.
Definici\ón de bipartito.
Conectitud.

Componentes conexas.
Repaso de BFS y DFS.
Algoritmo polinomial para determinar biparticidad
Propiedad: un grafo es bipartito si y solo
si no tiene ciclo impares.
Algoritmo Greedy de Coloreo.
Ejemplos de aplicación.
Ejemplo de que no siempre Greedy devuelve el número cromático.
Ejemplo de que tan mal puede dar.

==========
Very Important Theorem: (central para el proyecto)
Sea G=(V,E) un grafo cuyos vértices estan coloreados con un coloreo propio c con r colores {0,1,...,r-1}.

Sea P una permutación de los números 0,1,...,r-1.

Sea V[i]={x en V tal que c(x)=i}, i=0,1,...,r-1.

Ordenemos los vértices poniendo primero los vértices de V[P(0)], luego los de V[P(1)],etc, hasta $V[P(r-1)]
Entonces Greedy en ese orden coloreará G con r colores o menos.
==========
Propiedad: El número cromático es menor o igual que Delta +1.
Ejemplos donde se alcanza la cota.

Teorema de Brooks. (prueba solo para el caso no regular).

Teorema de los cinco colores.

BIBLIOGRAFÍA

BIBLIOGRAFÍA BÁSICA

Matemática Discreta. N. Biggs,1989

Applied Combinatorics. Roberts, 1989, Prentice-Hall.

Data Structures and Network Algorithms. R.E. Tarjan, 1983, Society
for Industrial and Applied Mathematics.

\Computers and Intractability: A Guide to the Theory of NP-completeness.
Garey and Johnson, 1979, Bell Telephone Laboratories.

.pdfs hechos por Daniel Penazzi y entregados a los alumnos.

Combinatorial Optimization: Algorithms and Complexity.
Papadimitriou-Steiglitz, 1998, Dover Publications.

BIBLIOGRAFÍA COMPLEMENTARIA

Applied Combinatorics. A. Tucker, 2nd Ed., 1984

Network Flows: Theory, Algorithms and Applications.
Ahoja-Magnani-Orlin, 1993, Prentice-Hal

EVALUACIÓN

FORMAS DE EVALUACIÓN

Habrá tres partes

1) Una parte teórica, sobre demostraciones de teoremas.

2) Una parte práctica sobre resolución de ejercicios.

3) Una parte de programación, que se tomará a lo largo del curso.

REGULARIDAD

Para regularizar los alumnos deberán aprobar el proyecto de programación. ("laboratorio")
De acuerdo con el inciso 3 del Art. 8° de la Ord. HCD N° 4/11 los alumnos deben aprobar el 60% de los trabajos pedidos. Como es un solo proyecto, deberan aprobarlo.

En cuanto a los parciales, (inciso 2 "aprobar al menos dos evaluaciones parciales o sus correspondientes recuperatorios", si es que se piden) seran 2 parciales asi que deberan aprobar los dos o sus respectivos recuperatorios.

PROMOCIÓN

La materia no se promociona.