

UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

(Universidad del Perú, Decana de América)

FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA



Escuela Académico Profesional de Ingeniería de Sistemas

ANÁLISIS Y DISEÑO DE ALGORITMOS

Integrantes:

- Fajardo Inca Miguel
- Garcia Licas Ronald
- Chacon Robles Jhony

Docente: Herminio Paucar Curasma

Github: <https://github.com/MiguelFajardoI/Grupo2-ADA-Unmsm-Fisi>

LIMA – PERÚ

2020

ÍNDICE

INTRODUCCIÓN	3
1. Marco Teórico	3
1.1 Topological Order:	4
1.2 Algoritmo:	4
1.3 Librerías:	4
2. Metodología de Desarrollo	5
3. Conclusiones y Recomendaciones	5
5. Aplicaciones y Grupos	7

INTRODUCCIÓN

La teoría de grafos, también llamada teoría de gráficas, es una rama de las matemáticas y las ciencias de la computación que estudia las propiedades de los grafos. Los grafos no deben ser confundidos con las gráficas, que es un término muy amplio. Formalmente, un grafo $G = (V, E)$ es una pareja ordenada en la que V es un conjunto no vacío de vértices y E es un conjunto de aristas.

La teoría de grafos tiene sus fundamentos en las matemáticas discretas y de las matemáticas aplicadas. Esta teoría requiere de diferentes conceptos de diversas áreas como combinatoria, álgebra, probabilidad, geometría de polígonos, aritmética y topología.

Actualmente ha tenido mayor influencia en el campo de la informática, las ciencias de la computación y telecomunicaciones. Debido a la gran cantidad de aplicaciones en la optimización de recorridos, procesos, flujos, algoritmos de búsquedas, entre otros, se generó toda una nueva teoría que se conoce como análisis de redes.

1. Marco Teórico

1.1 Topological Order:

Una ordenación topológica (topológico sort, topological, ordering, topsort o toposort).

De un grafo acíclico G dirigido es una ordenación lineal de todos los nodos de G que conserva la unión entre vértices del grafo G original. La condición que el grafo no contenga ciclos es importante, ya que no se puede obtener ordenación topológica de grafos que contengan ciclos.

Usualmente, para clarificar el concepto se suelen identificar los nodos con tareas a realizar en la que hay una precedencia a la hora de ejecutar dichas tareas. La ordenación topológica por tanto es una lista en orden lineal en que deben realizarse las tareas.

Para poder encontrar la ordenación topológica del grafo G deberemos aplicar una modificación del algoritmo de búsqueda en profundidad (DFS).

1.2 Algoritmo:

Los algoritmos habituales para la clasificación topológica tienen un tiempo de ejecución lineal en el número de nodos más el número de aristas, asintóticamente, $O(n + m)$.

Pseudocódigo:

```
ORDENACIÓN_TOPOLÓGICA(grafo G)
for each vértice  $u \in V[G]$  do
  estado[u] = NO_VISITADO
  padre[u] = NULL
  tiempo = 0
for each vértice  $u \in V[G]$  do
  if estado[u] = NO_VISITADO then
    TOPOLÓGICO-Visitar(u)
```

```
TOPOLÓGICO-Visitar(nodo u)
  estado[u] = VISITADO
  tiempo = tiempo + 1
  distancia[u] = tiempo
  for each  $v \in \text{Adyacencia}[u]$  do
    if estado[v] = NO_VISITADO then
      padre[v] = u
      TOPOLÓGICO-Visitar(v)
  estado[u] = TERMINADO
  tiempo = tiempo + 1
  finalización[u] = tiempo
  insertar (lista, u)
```

1.3 Librerías:

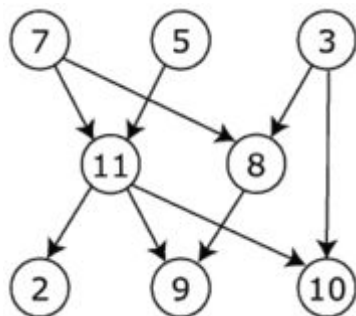
Para este proyecto hemos utilizado las librerías de:

- jQuery JavaScript Library v1.5.2
- jQuery UI 1.8.11

2. Metodología de Desarrollo

La utilidad del **orden topológico** en programación, es para una secuencia de tareas; los algoritmos de ordenamiento topológico fueron desarrollados para tareas secuenciales estas están representados por vértices, y con una **arista** desde x a y si la tarea x debe completarse antes que la tarea y empiece, por ejemplo, cuando se lava la ropa, la lavadora debe terminar antes de ponerla a secar la ropa. Por lo tanto, un **orden topológico** brinda un orden para ejecutar estas tareas.

El gráfico muestra que hay muchos tipos de ordenamiento posibles:



- 7, 5, 3, 11, 8, 2, 9, 10
- 3, 5, 7, 8, 11, 2, 9, 10
- 3, 7, 8, 5, 11, 10, 2, 9
- 5, 7, 3, 8, 11, 10, 9, 2
- 7, 5, 11, 3, 10, 8, 9, 2
- 7, 5, 11, 2, 3, 8, 9, 10

¿Cómo afrontamos el problema?


Primero tenemos que identificar qué nodos tienen entradas y cuántas entradas tienen, esto nos muestra a los nodos con ninguna entrada (entradas = 0).

Creamos una cola donde ingresamos los nodos con entradas 0 para su análisis y permanencia en la solución final.

El análisis consta de ver qué salidas tiene y eliminarlas, después actualizamos la cola con los nuevos vértices de entrada = 0.

Y vamos haciendo esto hasta que la cola de vértices con entradas = 0 quedé vacía.

Prototipo de la Aplicación:

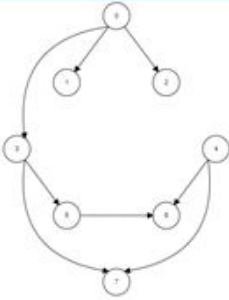


Facultad de Ingeniería de Sistemas e Informática
(Análisis y Diseño de Algoritmo)

De Topological Sort | New Graph | Small Graph | Representación del gráfico

Integers

0
1
2
3
4
5
6
7



Animation Controls

Stop Back | Stop End | Pause | Step Forward | Skip Forward

Animation Speed

W: 1000 H: 500 Change Canvas Size Move Controls

Pseudocódigo:

```
ORDENACIÓN_TOPOLOGICA(graf G)
for each vértice u ∈ V(G) do
    estado[u] = NO_VISITADO
    padre[u] = NULL
    tiempo = 0
for each vértice u ∈ V(G) do
    if estado[u] = NO_VISITADO then
        TOPOLOGICO-Visitar(u)

TOPOLOGICO-Visitar(nodo u)
    estado[u] = VISITADO
    tiempo = tiempo + 1
    distancia[u] = tiempo
    for each v ∈ Adyacencia[u] do
        if estado[v] = NO_VISITADO then
            padre[v] = u
            TOPOLOGICO-Visitar(v)
    estado[u] = TERMINADO
    tiempo = tiempo - 1
    finalizacion[u] = tiempo
    insertar (lista, u)
```

3. Conclusiones y Recomendaciones

- Al concluir este proyecto comprendimos que para ejecutar o correr un programa en una página web no solo es necesario el código o algoritmo que vamos a desarrollar, sino que es necesaria una interfaz gráfica ya que hoy en día casi ninguna página está basada solo en texto.
- Como recomendación podríamos decirles que si quieren hacer animaciones o efectos de una forma un poco más sencilla pueden hacer uso de la librería JQuery.

4. Referencias:

- *Kahn's algorithm for Topological Sorting*. (2020, 31 mayo). GeeksforGeeks.
<https://www.geeksforgeeks.org/topological-sorting-indegree-based-solution/>
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed. ed.). MIT Press.
- *Curso Básico de Javascript desde 0 - Introducción*. (2015, 3 agosto). [Video]. YouTube.
https://www.youtube.com/watch?v=xnWtGNiG2lg&list=PLhSj3UTs2_yVC0iaCGf16glrrfXuiSd0G

5. Aplicaciones y Grupos

La aplicación canónica del orden topológico es en programación, una secuencia de tareas; en el contexto de la técnica "PERT" (Técnica de Revisión y Evaluación de Programas del inglés) de programación en gestión de proyectos Las tareas están representados por vértices, y hay un arco (o arista) desde x a y si la tarea x debe completarse antes que la tarea y comience.

- Cuando se lava la ropa, la lavadora debe terminar antes de ponerla a secar. Entonces, un **orden topológico** brinda un orden para ejecutar las tareas.
- Ø En la vida diaria lo podríamos aplicar en cosas simples por ejemplo, organizar los recibos de los servicios a pagar, ya que estos llegan en diferentes días, y difieren en precio y vencimiento, y al aplicarle este algoritmo, podríamos ordenarlos en orden de prioridad de vencimiento, para irlos pagando ordenadamente, sin que se nos pasen y sin que gastemos el dinero necesario para pagarlos.

- Puede ser aplicable a la representación de las fases de un proyecto, donde los vértices representan tareas, y las aristas relaciones temporales a ellas.
- Evaluación en la fase semántica de un compilador
- Sirve para encontrar algún ciclo en un grafo e identificarlo
- Para encontrar algún problema o defecto de cierto algoritmo que se ordene topológicamente.