

Proyecto # 3: Kevin el encargado



Laura Victoria Riera Pérez
Marié del Valle Reyes

Cuarto año. Ciencias de la Computación.
Facultad de Matemática y Computación, Universidad de La Habana, Cuba

7 de junio de 2023

I. REPOSITORIO DEL PROYECTO

<https://github.com/computer-science-crows/algorithms-design-and-analysis>

II. DEFINICIÓN INICIAL DEL PROBLEMA

Kevin ha sido puesto al frente de la comisión de la facultad que elegirá las fechas de las pruebas de los k cursos que se dan en la facultad.

Cada curso tiene una cantidad de pruebas determinadas que quiere poner, y propone para esto, por ejemplo, los días $\{ 17, 34, 65 \text{ y } 87 \}$ del curso escolar, si vemos a este como una sucesión de días en los que se imparten clases. Para mostrarse flexibles, los cursos a veces elaboran más de una propuesta incluso.

Por un problema de desorganización las propuestas se regaron y ahora no se sabe que curso propuso que propuesta, pero ya Kevin esta cansado de tanta gestión. Kevin quiere elegir k propuestas que ninguna quiera poner pruebas el mismo día que las otras, así supone que todo el mundo estará contento, ayude a Kevin.

III. DEFINICIÓN EN TÉRMINOS MATEMÁTICO - COMPUTACIONALES

I. Problema de Selección de Actividades

IV. SOLUCIONES IMPLEMENTADAS

I. Backtrack

Como primera solución al problema fue implementado un *backtrack*. Esta es una solución correcta, ya que prueba todas las combinaciones de posibles carreteras y se queda con la que más ganancia aporte, pero muy ineficiente $O(2^m)$. En una computadora de 32GB de RAM, intel core i7-11na generación, se puede resolver para una cantidad máxima 5 ciudades con 5 aristas. Dicha solución puede ser encontrada en `src/solutions/backtrack_solution.py`.

II. Máximo conjunto independiente

III. Activity selection problem algorithm

IV. Algoritmo genético

V. Solución

v.1. Preliminares

v.2. Propuesta de solución

v.3. Complejidad Temporal

v.4. Complejidad Espacial

V. GENERADOR DE CASOS DE PRUEBA

En `src/app/generator.py` fue implementado un generador, el cual recibe una cantidad s de muestras a producir, genera valores random con el formato de entrada de los algoritmos implementados, halla la solución óptima con *backtrack* y las guarda en `json/test_cases.json`. Se generaron 3000 casos de prueba.

VI. TESTER

En `src/app/tester.py` fue implementado un tester, que recibe una función y prueba el desempeño de la misma en cuanto a si obtuvo la solución óptima o no, y el tiempo que demoró en hacerlo, comparando con los casos de prueba obtenidos con el generador. Además, estos resultados se guardan en un `.json` con el nombre de la función en la carpeta tests. La solución implementada fue testeada para todos los casos de prueba generados y puede encontrarse en `json/tests/corruption_strategy_solution.json`.

VII. COMPARACIÓN DE SOLUCIONES IMPLEMENTADAS

REFERENCIAS

- [1] Cormen, Thomas H. y otros. *Introduction to Algorithms*. The MIT Press. 4ta Edición. Cambridge, Massachusetts. 2022.