Universidad Nacional Autónoma de México

Facultad de Ciencias

Complejidad Computacional

Semestre 2024-2

Programa 01:

Algoritmos No Deterministicos

Deloya Andrade Ana Valeria

317277582

Ruta Más Corta

a. Dar su forma canónica.

Dada una gráfica no dirigida G = (V,E), u, v vértices en V, y k entero positivo, ¿existe una uv-trayectoria en G de distancia menor a k?

b. Diseñar un algoritmo no-determinístico polinomial.

- **Fase Adivinadora:**

  Comienza por revisar una a una las aristas de la gráfica G, y por medio de una variable que va a arrojar un valor aleatorio entre 0 y 1 es que se decide si incluir o no a la arista en la trayectoria.

- **Fase Verificadora:**

  La lógica para la fase verificadora es la siguiente, tenemos una trayectoria que nos dio la fase adivinadora, por ejemplo '5,1', '1,2', '2,6' que podemos ver como 5,1,1,2,2,6

  Si nos ponemos a analizar, en general lo que queremos es comparar el segundo y tercer elemento de la trayectoria, de ahí seguir con el cuarto y quinto, y así sucesivamente.

  Por lo que el primer elemento no lo vamos a ocupar para esta comparación y tampoco el último, así que comienza la comparación a partir del segundo elemento.

  Si al comparar dos vértices resultan ser iguales, contamos esas dos aristas y sumamos dos al conteo de la distancia. Y seguimos pero ahora comparando los dos vértices que siguen.

  Si llega a pasar que la comparación resulta no ser exitosa, dejamos de hacer comparaciones y regresamos $false$

Pero en caso de que sólo haya comparaciones exitosas:

En el aspecto del conteo de la distancia, nos piden que la distancia de la uv-trayectoria sea el numero de vértices sin contar u pero sí v.

Como no ocupamos v en las comparaciones, jamás sería contada, por lo que la distancia la empezamos a contar a partir de 1 y no de 0.

Como puede pasar que nuestra primera comparación no sea exitosa, la variable que cuenta la distancia sería 1. Siendo éste el porqué al momento de verificar si la distancia es menor a k también debemos de ver que sea diferente de 1.

Regresamos $true$ si se cumplen ambas condiciones: que sea menor a k y diferente de 1. En caso contrario regresamos $false$

3-SAT

a. Dar su forma canónica.

Sea una expresión booleana B en su Forma Normal Conjuntiva con 3 literales para cada una de sus clausulas, ¿existe una asignación de valores para B que la satisfaga?

b. Diseñar un algoritmo no-determinístico polinomial.

- **Fase Adivinadora:**

  Comienza por asignar un valor de verdad a cada variable que vamos a tener en la fórmula booleana: x, y, z. Este valor de verdad se asigna de manera aleatoria, teniendo 1 como $true$ y 0 como $false$.

  Una vez teniendo qué valores serán para cada variable, recorremos la fórmula booleana y uno a uno vamos asignando los valores correspondientes. Todo esto lo hacemos en una función.

- **Fase Verificadora:**

  Para verificar si la asignación satisface la fórmula booleana, comenzamos por encargarnos de los *not*. Como las variables de la fórmula ya tienen un valor *true* o *false* asignado, vamos a tener una función que va a recorrer esta fórmula y si se encuentra con un -1 lo cambia por un 0 mientras que si encuentra un -0 lo cambiará por un 1.

  Posteriormente, tendremos otra función para resolver las operaciones *or* de acuerdo a la tabla de valores de verdad, donde sólo tenemos valores *false* cuando todas nuestras variables son *false*. En el resto de los casos tenemos el valor *true*. Entonces sustituímos en la fórmula por 1 o 0 según sea el caso.

  Finalmente habrá otra función para resolver las operaciones *and* de acuerdo a la tabla de valores de verdad donde sólo tenemos valor *true* cuando todas nuestras variables son *true*. En el resto de los casos es *false*. Entonces sustituímos en la fórmula por 1 o 0 según sea el caso.

  Al final de esta última función, se devuelve un valor booleano con el que sabremos si se satisface o no la fórmula con la asignación de valores dada en la fase adivinadora, si es 1 se satisface y si es 0 no se satisface.