Computabilità ed Algoritmi
Esempio di Esame di Algoritmi
25 Maggio 2018

Tempo a disposizione: 1 h 30 min

1. Considerate il problema descritto nel seguito. Trasformate l'input in un grafo e applicate uno degli algoritmi visti a lezione per risolverlo. Fornite le seguenti informazioni:

- Quali sono i vertici? Cosa rappresenta ciascun vertice?
- Quali sono gli archi? Sono orientati o non orientati?
- Se i vertici e/o gli archi hanno dei valori associati, quali sono questi valori?
- Quale problema si deve risolvere sul grafo?
- Quale algoritmo si sta usando per risolvere il problema?
- Qual è il tempo di esecuzione dell'intero algoritmo, *incluso* il tempo di creazione del grafo, *in funzione dei parametri di input originali*?

Sia G un grafo non orientato. Supponiamo di iniziare con due monete disposte su due vertici arbitrariamente scelti di G. Ad ogni passo, ogni moneta *deve* spostarsi su un vertice adiacente.

Descrivere e analizzare un algoritmo efficiente per calcolare il numero minimo di passaggi per raggiungere una configurazione in cui entrambe le monete si trovano sullo stesso vertice o per segnalare che tale configurazione non è raggiungibile.

L'input dell'algoritmo consiste in un grafo $G = (V, E)$ e due vertici $u, v \in V$ (che possono o non possono essere distinti).

2. Hai appena scoperto il tuo migliore amico della scuola elementare su Twitbook. Volete incontrarvi il prima possibile, ma vivete in due diverse città distanti tra di loro. Per ridurre al minimo i tempi di viaggio, stabilite di incontrarvi in una città intermedia, e poi salite contemporaneamente in auto per guidare l'uno verso l'altro. Ma dove dovreste incontrarvi esattamente?

Supponente di avere un grafo pesato $G = (V, E)$, in cui i vertici V rappresentano le città e gli archi E rappresentano le strade che collegano direttamente le città. Ogni arco $\{u, v\}$ ha un peso $w(u, v)$ uguale al tempo richiesto per viaggiare tra le due città. Vi viene fornito anche un vertice p, che rappresenta la tua posizione di partenza, e un vertice q, che rappresenta la posizione di partenza del tuo amico.

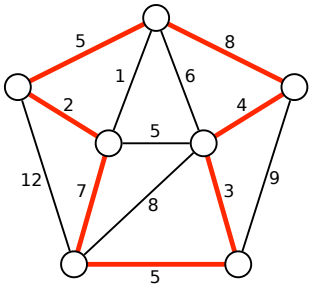
Descrivi e analizza un algoritmo per trovare il vertice intermedio che consente a te e al tuo amico di incontrarvi il prima possibile, assumendo che partiate entrambi da casa *nello stesso momento*.

3. Dimostrare che un problema X è NP-hard richiede diversi passaggi:

- Scegli un problema Y che sai essere NP-hard (perché l'hai visto a lezione).
- Descrivi un algoritmo per risolvere Y usando un algoritmo per X come subroutine. Tipicamente questo algoritmo ha la seguente forma: data un'istanza di Y, trasformala in un'istanza di X, quindi chiama l'algoritmo magico black-box per X.
- Dimostra che l'algoritmo è corretto. Ciò richiede sempre due passaggi separati, che di solito hanno la seguente forma:
  - Dimostra che il tuo algoritmo trasforma istanze "buone" di Y in istanze "buone" di X.
  - Dimostra che il tuo algoritmo trasforma istanze "cattive" di Y in istanze "cattive" di X. Equivalentemente: Dimostra che se la tua trasformazione produce un'istanza "buona" di X, allora era partita da un'istanza "buona" di Y.
- Mostra che il tuo algoritmo per Y funziona in tempo polinomiale, a meno della chiamata (o delle chiamate) all'algoritmo magico black-box per X. (Questo di solito è banale.)

Un circuito Hamiltoniano in un grafo G è un ciclo che attraversa ogni vertice di G esattamente una volta. Stabilire se un grafo arbitrario contiene un circuito Hamiltoniano è un problema NP-hard. Anche stabilire se il grafo contiene un *cammino* Hamiltoniano è NP-hard.

Sia G un grafo non orientato e pesato. Un circuito Hamiltoniano di G è *pesante* se il peso totale degli archi nel ciclo è almeno la metà del peso totale di tutti gli archi di G. Dimostrare che stabilire se un grafo contiene un ciclo Hamiltoniano pesante è NP-hard.



Un ciclo Hamiltoniano pesante. Il ciclo ha un peso totale 34; il grafo ha un peso totale 67.