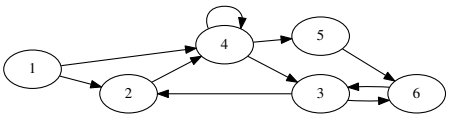
Programmazione Funzionale – Giugno 2021

Nota: è indispensabile specificare il tipo e dare una descrizione dichiarativa di ogni funzione ausiliaria utilizzata (anche locale), altrimenti non verrà presa in considerazione (ad eccezione delle funzioni il cui tipo e specifica sono già dati nel testo).

1. Dato un grafo G, un nodo sorgente *start*, un nodo obiettivo *goal* e un limite *depth*, la *ricerca in profondità limitata* riporta, se esiste, un cammino in G da *start* a *goal* di lunghezza non maggiore di *depth* (la lunghezza di un cammino è uguale al numero degli archi che lo compongono). La ricerca avviene in profondità e fallisce se un tale cammino non esiste.

   Ad esempio, nel grafo sotto rappresentato, la ricerca in profondità limitata a partire dal nodo 1 con obiettivo 6 e limite $depth \leq 2$ fallisce; se invece $depth > 2$, la ricerca ha successo. Con $depth = 3$ verrà riportato uno dei cammini [1;4;3;6] o [1;4;5;6]. Se $depth > 5$, la ricerca può riportare, ad esempio, anche il cammino [1;2;4;3;6], o un cammino con cicli, come ad esempio [1;4;3;2;4;5;6].

   Definire un tipo di dati α `graph` per la rappresentazione di grafi orientati mediante liste di archi e definire una funzione

   `depth_limited:` α `graph` $\rightarrow \alpha \rightarrow \alpha \rightarrow$ `int` $\rightarrow \alpha$ `list`

   tale che `depth_limited g start goal depth` riporti, se esiste, un cammino nel grafo `g` dal nodo `start` al nodo `goal` la cui lunghezza non sia maggiore di `depth`.

2. La ricerca di un cammino *per approfondimenti successivi* consiste in una reiterazione della ricerca in profondità limitata, incrementando ad ogni iterazione il limite *depth*: innanzitutto si cerca un cammino di lunghezza 0, poi, se questa ricerca fallisce, si cerca un cammino di lunghezza 1, e così via, fino a che la ricerca ha successo oppure viene soddisfatta una determinata condizione di arresto.

   Utilizzando la funzione principale del punto precedente, definire una funzione

   `path:` α `graph` $\rightarrow \alpha \rightarrow \alpha \rightarrow$ `int` $\rightarrow \alpha$ `list`

   che implementi la ricerca per approfondimenti successivi con limite di profondità massimo prefissato: `path g start goal maxdepth` reitera la ricerca in profondità limitata fino a che il limite *depth* non supera il valore `maxdepth`.

3. Cosa significa che nei linguaggio funzionali le funzioni sono "oggetti di prima classe"? Illustrare questo concetto anche mediante esempi, facendo riferimento ad OCaml.