

# Teoria dei grafi

DP su alberi e DAG

Lorenzo Ferrari, Davide Bartoli

January 18, 2023

# Table of contents

## Parallelismo tra DP e DAG

- DP e DAG

- DP e albero

## Problemi

- Problemi

## Parallelismo tra DP e DAG

Nelle lezioni precedenti abbiamo detto che “tutto è grafo”, troviamo grafi anche dove non ce li aspetteremmo.

# Parallelismo tra DP e DAG

Nelle lezioni precedenti abbiamo detto che “tutto è grafo”, troviamo grafi anche dove non ce li aspetteremmo.

Consideriamo una generica dp

- ▶ vediamo ogni stato come un nodo
- ▶ se lo stato  $b$  dipende dallo stato  $a$ , rappresentiamo questa dipendenza con un arco orientato  $(a, b)$
- ▶ per calcolare la dp, visitiamo i nodi in **ordinamento topologico**

# Parallelismo tra DP e DAG

## Ordinamento topologico

### Ordinamento topologico

Per **ordinamento topologico** (in inglese *topological sort* o *toposort*), si intende un ordinamento dei nodi di un grafo diretto in modo che ogni nodo viene prima di tutti i nodi collegati ai suoi archi uscenti.

---

<sup>1</sup>generalmente ne esistono più di uno

# Parallelismo tra DP e DAG

## Ordinamento topologico

### Ordinamento topologico

Per **ordinamento topologico** (in inglese *topological sort* o *toposort*), si intende un ordinamento dei nodi di un grafo diretto in modo che ogni nodo viene prima di tutti i nodi collegati ai suoi archi uscenti.

Un grafo è un DAG **se e solo se** esiste un suo ordinamento topologico.

---

<sup>1</sup>generalmente ne esistono più di uno

# Parallelismo tra DP e DAG

## Ordinamento topologico

### Ordinamento topologico

Per **ordinamento topologico** (in inglese *topological sort* o *toposort*), si intende un ordinamento dei nodi di un grafo diretto in modo che ogni nodo viene prima di tutti i nodi collegati ai suoi archi uscenti.

Un grafo è un DAG **se e solo se** esiste un suo ordinamento topologico.

A seconda delle implementazioni della nostra dp, potrebbe essere necessario trovare un toposort<sup>1</sup>.

---

<sup>1</sup>generalmente ne esistono più di uno

# Parallelismo tra DP e DAG

## Ordinamento topologico

### Ordinamento topologico

Per **ordinamento topologico** (in inglese *topological sort* o *toposort*), si intende un ordinamento dei nodi di un grafo diretto in modo che ogni nodo viene prima di tutti i nodi collegati ai suoi archi uscenti.

Un grafo è un DAG **se e solo se** esiste un suo ordinamento topologico.

A seconda delle implementazioni della nostra dp, potrebbe essere necessario trovare un toposort<sup>1</sup>.

Se sappiamo il grafo è un dag, si può facilmente trovare un toposort con una dfs.

---

<sup>1</sup>generalmente ne esistono più di uno



# Parallelismo tra DP e DAG

## Fibonacci

Prendiamo per esempio la dp dei numeri di Fibonacci

$$F_n = \begin{cases} 1 & \text{per } 0 \leq n \leq 1 \\ F_{n-1} + F_{n-2} & \text{per } n \geq 2 \end{cases}$$

# Parallelismo tra DP e DAG

## Fibonacci

Prendiamo per esempio la dp dei numeri di Fibonacci

$$F_n = \begin{cases} 1 & \text{per } 0 \leq n \leq 1 \\ F_{n-1} + F_{n-2} & \text{per } n \geq 2 \end{cases}$$

Il DAG equivalente è il seguente



# Parallelismo tra DP e DAG

Se ogni dp è un DAG, allora fare una dp su un DAG è molto più naturale di quel che sembra.


Semplicemente le relazioni di dipendenza tra stati sono arbitrarie e vengono date in input.

## Esempio

Dato un DAG, contare i percorsi da 0 a  $n - 1$

# Contare percorsi

## Implementazione



```
vector<int> adj[100000];
vector<bool> vis(100000, false);
vector<int> memo(100000);
int conta(int pos) {
    if (pos == N - 1) return 1;
    if (vis[pos]) return memo[pos];
    vis[pos] = true;
    int tot = 0;
    for (int nodo : adj[pos]) {
        tot += conta(nodo);
    }
    return tot; // % mod;
}

ans = conta(0);
```

## DP su albero

Possiamo applicare questa idea anche a problemi su alberi.

## DP su albero

Possiamo applicare questa idea anche a problemi su alberi. Nonostante gli alberi generalmente non siano DAG (dato che gli archi sono bidirezionali), possiamo sempre trovare un DAG equivalente, fissando un nodo radice e orientando gli archi verso i figli.

## DP su albero

Possiamo applicare questa idea anche a problemi su alberi. Nonostante gli alberi generalmente non siano DAG (dato che gli archi sono bidirezionali), possiamo sempre trovare un DAG equivalente, fissando un nodo radice e orientando gli archi verso i figli.

Questa idea ci permette di risolvere facilmente problemi su alberi, come ad esempio calcolare la dimensione di ogni sottoalbero.

# Problemi

<https://cses.fi/problemset/task/1674>

<https://cses.fi/problemset/task/1130>

<https://cses.fi/problemset/task/1131>

<https://training.olinfo.it/#/task/traffic/statement>

[https://training.olinfo.it/#/task/ois\\_magicforest/statement](https://training.olinfo.it/#/task/ois_magicforest/statement)

[https://training.olinfo.it/#/task/ois\\_xmastree/statement](https://training.olinfo.it/#/task/ois_xmastree/statement)

[https://training.olinfo.it/#/task/ois\\_picarats/statement](https://training.olinfo.it/#/task/ois_picarats/statement)

[https://training.olinfo.it/#/task/preoii\\_catene/statement](https://training.olinfo.it/#/task/preoii_catene/statement)



Fateci sapere come stanno andando le lezioni

<https://forms.gle/WGPqv2ctQ34b4tz9A>