

# Algoritmi i strukture podataka

11 Grafovi

Katedra za informatiku, Fakultet tehničkih nauka, Novi Sad

2023

# Osnovni pojmovi

- **graf** je par  $(V, E)$
- $V$  je set **čvorova** (*vertices*)
- $E$  je skup **grana** (*edges*)
- čvorovi i grane čuvaju **elemente**

# Osnovni pojmovi

- **Usmerena grana**
  - uređeni par čvorova  $(u, v)$
  - prvi čvor  $u$  je polazište
  - drugi čvor  $v$  je odredište
- **Neusmerena grana**
  - neuređeni par čvorova  $(u, v)$
- **Usmereni graf**
  - sve grane su usmerene
- **Neusmereni graf**
  - sve grane su neusmerene

# Implementacija

- Poželjno je kroz jednu implementaciju podržati kreiranje i usmerenog i neusmerenog grafa
- Ukoliko nemate ideju kako to da postignete, implementirajte odvojeno usmeren i neusmeren graf pa pokušajte naknadno da povežete ove dve implementacije
- Koristićemo implementaciju pomoću dva rečnika
- Graf treba da podrži sledeće metode:

# Metode klase Graph

<code>vertex_count()</code>	broj čvorova
<code>vertices()</code>	lista svih čvorova
<code>edge_count()</code>	broj grana
<code>edges()</code>	lista svih grana
<code>get_edge(u,v)</code>	vraća granu između u i v ako postoji, inače None
<code>degree(v,out=True)</code>	broj izlaznih/ulaznih grana iz v
<code>incident_edges(v,out=True)</code>	lista izlaznih/ulaznih grana iz v
<code>insert_vertex(x=None)</code>	dodaj novi čvor sa sadržajem x
<code>insert_edge(u,v,x=None)</code>	dodaj novu granu od u ka v sa sadržajem x
<code>remove_vertex(v)</code>	ukloni čvor v i sve vezane grane
<code>remove_edge(e)</code>	ukloni granu e

# Formiranje grafa

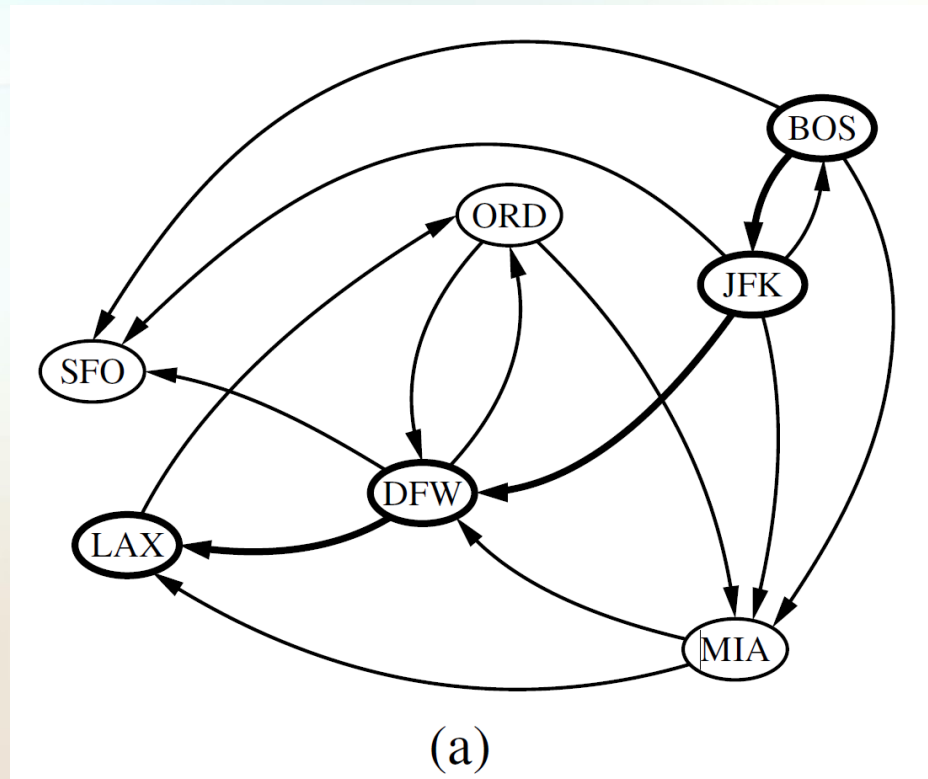
- Koji redosled operacija je potreban kako bi se kreirao graf od uređenih parova koji odgovaraju granama?
- Npr.
- $E = ($   
('SFO', 'LAX', 337), ('SFO', 'BOS', 2704), ('SFO', 'ORD', 1846)  
 $)$
- Moramo prvi izvući podatke o čvorovima
- Dodajemo (nepovezane) čvorove u graf
- Zatim, dodajemo grane

# Modul primeri\_graf

- Kada implementiramo klasu graf, kako da proverimo da li je ispravna?
- Za olakšanje rada, dostupan je modul primeri\_graf
- Modul primeri\_graf koristi vašu implementaciju grafa (kroz import)
- Sadrži metodu graph\_from\_edgelist koja na osnovu uređenih n-torki formira graf.
- Metode figure\_14\_3, figure\_14\_9, figure\_14\_12, figure\_14\_14 i figure\_14\_15 sadrže opise grafova koji su prikazani na narednim slajdovima

# Grafovi primeri

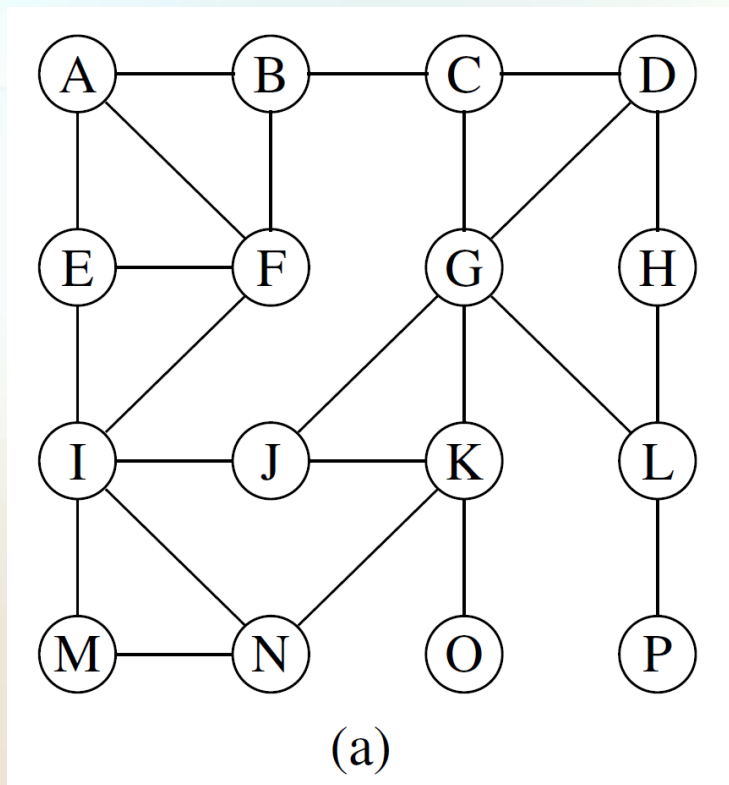
- Graf 14.3.





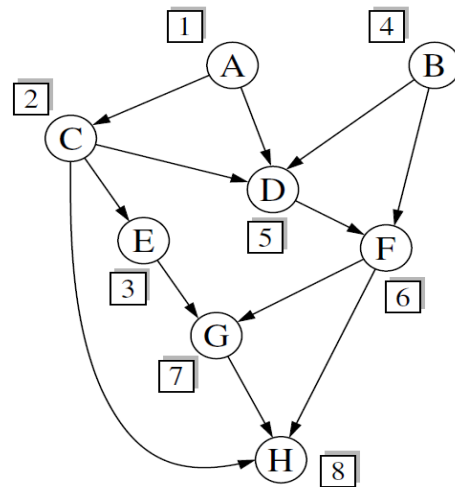
# Grafovi primeri

- Graf 14.9.

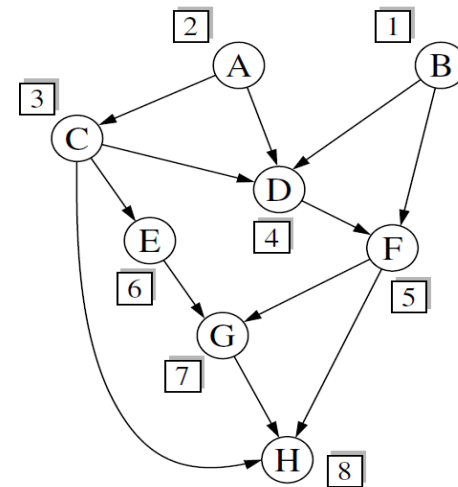


# Grafovi primeri

- Graf 14.12.



(a)

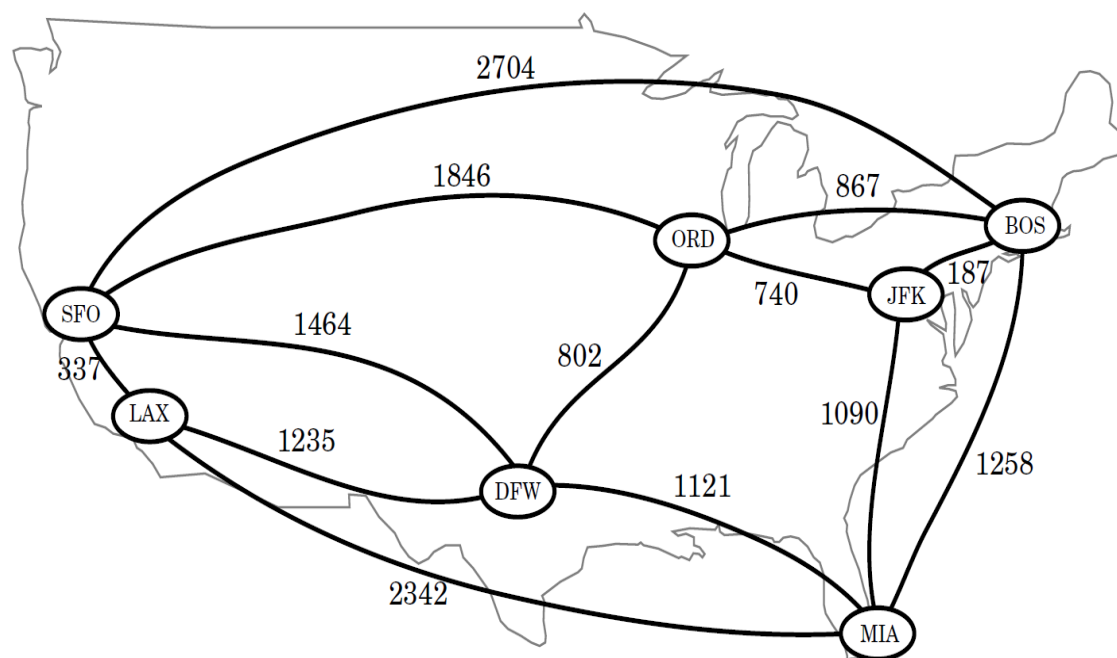


(b)

**Figure 14.12:** Two topological orderings of the same acyclic directed graph.

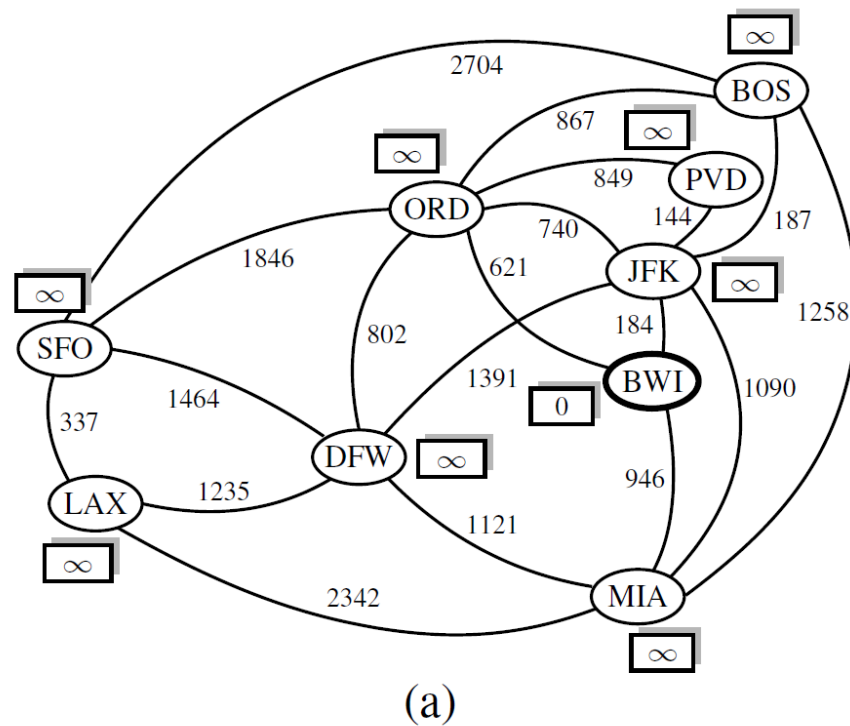
# Grafovi primeri

- Graf 14.14.



# Grafovi primeri

- Graf 14.15.



# Zadatak 1

- Implementirati klasu Graph.

## Zadatak 2

- Implementirajte DFS algoritam.
- ***Napomena:*** Za testiranje upotrebite [primeri\\_graf.py](#)

## Zadatak 3

- Implementirajte BFS za graf.
- ***Napomena:*** Za testiranje upotrebite [primeri\\_graf.py](#)