ДАА

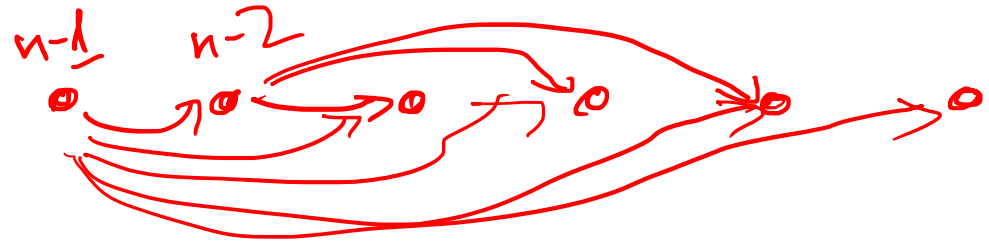
упр.№11, 5 юни 2020г.

Hello, world!

DAGs

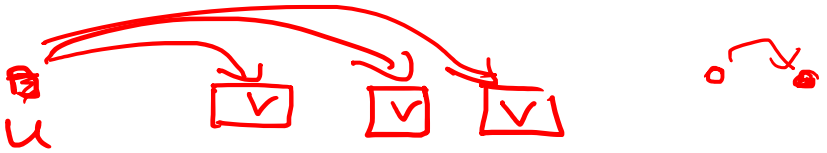
- дефиниция Directed Acyclic Graphs
- кога се срещат
- ползи:

Зад.4. Да се намери максималния брой ребра в DAG с n върха



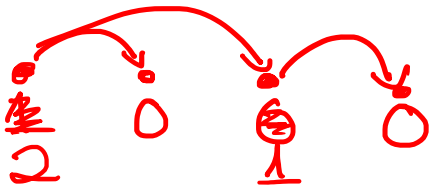
$$\Rightarrow \text{макс. брой ребра} = \sum_{i=1}^{n-1} i = \frac{(n-1)n}{2} = \Theta(n^2)$$

Зад.5. Да се намери най-дългият път между някои два върха в DAG → LP2DAG



LPDAG(G(V,E) – DAG)

1. A[1..n] ← TopoSort(G)
2. l[1..n] ← l[i] ще даде дълж. на най-дългия път, започващ от връх i
3. foreach u in A in reverse order ←
4. l[u] ← 0
5. foreach v in Adj(u)
6. l[u] = max{l[u], l[v]+1}
7. намираме за кое u l[u] е макс.
8. за Vi имаме да намерим за кой съсед v



w(u,v) → за тегловни

$T(n,m) = \underset{\text{TopoSort}}{\tilde{\Theta}(n+m)} + \tilde{\Theta}(n+m) + \underset{\min}{\tilde{\Theta}(n)} + O(n+m)$ бързо.

$M(n,m) = \underset{\text{TopoSort}}{\tilde{\Theta}(n)} + \underset{l}{\tilde{\Theta}(n)} = \tilde{\Theta}(n)$

$l[u] = l[v] + 1$

Зад.6. Да се докаже, че в DAG има НР т.с.т.к. графът има единствена топологична наредба → за вкъщи

подредена последователно

Зад.7. Даден е речник от думи с мистериозни символи. Да се намери дали в него има противоречия, или ако не – да се състави една възможна азбука от тези символи

abc	abc	7*&++
acc	acc	a4%%
baby	bdj	a4&%
bdj	bad	&~~xx
cafe	ccc	&~af
	damn	~oops
→ Няма противоречия	→ Има противоречия	→ **?**

За минималните покриващи дървета (MST)

- алгоритъм на Крускал

Започва от гора от върхове и ги обединява, използвайки Union-Find и обработвайки ребрата по тегло

$$T(n,m) = \tilde{\Theta}(m \lg m) + O(m \cdot \underbrace{\alpha(n)}_{<5})$$

$$M(n,m) = \tilde{\Theta}(1) + \tilde{\Theta}(n) = \tilde{\Theta}(n)$$

- алгоритъм на Прим

Започва от произв. връх и също добавя ребра, докато не включи всички други върхове

⇒ като импл. е 1:1 с алг. на Дийкстра

...ами за второто по големина MST?

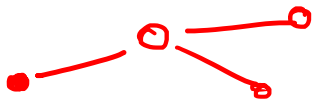
→ има не много сложен алгоритъм, но за $O(n^2)$

Покриващи дървета с минимално най-тежко ребро (MBST)

minimum bottleneck spanning tree

- линеен не много сложен алгоритъм за намирането им

- всяко MST е MBST, но не и обратното ➔ не можем да използваме този алгоритъм за бързо намиране на MST



За откриването на най-къси (леки) пътища

- алгоритъм на Дийкстра
  - има специални варианти за специални графи

Инициализира масив (приоритетна опашка) с най-добрите (засега) тегла от началния връх до всеки друг

не работи с < 0 тегла

Binary

$$T(n,m) = n.T_{em} + m\,T_{dk} = \begin{cases} n.O(\lg n) + m.O(\lg n) = O((n+m)\lg n) \\ n.O(\lg n) + m.O(1) = O(m + n\lg n) \\ n.O(\lg n) + m\,O(\lg n) = O((n+m)\lg n) \end{cases}$$

FibHeap

PairingHeap

ако теглата са ограничени и целочислени, има вариации за линейно време

За откриването на най-къси (леки) пътища

- алгоритъм на Белман-Форд → справя се с ребра с <0 тегла; отриц. цикли намира; $T(n,m) = O(n \cdot m)$
- алгоритъм на Флойд-Уоршал → еднокр. намира (теглата) пътищата от всеки до всеки; $T(n,m) = \Theta(n^3)$ n^2

Зад.8. **n** студенти на изпит, знаем кой на кого е възможно да подскаже директно (опасно или безопасно). За дадени двама студенти A и B да се намери възможно най-безопасна последователност от студенти за подсказване.

Дийкстра в/у граф с n върха (съотв. на n-те студенти)

а за ребрата

- ако двама студенти не могат да си подскажат директно → няма ребро м/у съответстващите им върхове
- ако могат → ще има ребро с тегло 1, ако е опасно, и 0 ако е безопасно