

# Algoritmy a grafy 1 (BI-AG1), Cvičení č. 3

## Stromy, prohledávání do šířky

Paralelka 104, Úterý 16:15-17:45

Cvičící: Šimon Lomič  
[lomicsim@fit.cvut.cz](mailto:lomicsim@fit.cvut.cz)

Informace: [lomicsim.github.io](https://github.com/lomicsim)

Fakulta informačních technologií  
České vysoké učení technické v Praze  
<https://courses.fit.cvut.cz/BI-AG1>



(Verze dokumentu: 29. 10. 2018 19:38)

# Opakování z přednášky – Stromy

- **Strom** je souvislý graf, který neobsahuje žádnou kružnici (**acyklický**).
- **Les** je libovolný acyklický graf.
- **List** grafu je libovolný vrchol stupně 1.
- Každý strom  $T$  s aspoň 2 vrcholy obsahuje aspoň dva listy.
- Nechť  $G = (V, E)$  je graf na aspoň 2 vrcholech a nechť  $v \in V$  je list. Pak  $G$  je strom  $\Leftrightarrow G - v$  je strom.
- Nechť  $G = (V, E)$  je graf. Pak následující tvrzení jsou ekvivalentní:
  - 1  $G$  je strom.
  - 2 Pro každé dva vrcholy  $u, v \in V$  existuje právě jedna  $u$ - $v$ -cesta.
  - 3  $G$  je souvislý a vynecháním libovolné hrany vznikne nesouvislý graf.
  - 4  $G$  je souvislý a  $|V| = |E| + 1$ .

**Cvičení:** Kolik listů může maximálně, resp. minimálně mít strom na  $n$  vrcholech ( $n \geq 2$ )?

## 3.1 Stromy

- (a) Necht  $T$  je neorientovaný strom, v němž je maximální stupeň vrcholu  $\Delta(T)$ . Ukažte, že strom  $T$  má nejméně  $\Delta(T)$  listů.
- (b) Necht  $d_1, \dots, d_n$  jsou kladná celá čísla. Dokažte následující ekvivalenci: Existuje strom na  $n$  vrcholech se stupni  $d_1, \dots, d_n$  právě tehdy, když  $\sum_{i=1}^n d_i = 2n - 2$ . **(0.5b)**

## 3.2 Algoritmy DFS a BFS

Algoritmus **DFS** ( $G, u$ ):

```
(1) visited = {}  
(2) stack = new Stack({u})  
(3) while stack.notEmpty()  
(4)      $v = \text{stack.pop}()$ ;  
(5)     if  $v \in \text{visited}$ :  
(6)         continue;  
(7)     visited.insert( $v$ )  
(8)     for  $w \in N(v)$   
(9)         stack.push( $w$ )
```

Algoritmus **BFS** ( $G, u$ ):

```
(1) visited = {}  
(2) queue = new Queue({u})  
(3) while queue.notEmpty()  
(4)      $v = \text{queue.pop}()$ ;  
(5)     if  $v \in \text{visited}$ :  
(6)         continue;  
(7)     visited.insert( $v$ )  
(8)     for  $w \in N(v)$   
(9)         queue.push( $w$ )
```

### Cvičení:

- (a) Proč nelze použít algoritmus DFS k nalezení nejkratší  $u$ - $v$ -cesty?
- (b) Modifikujte iterační verzi algoritmu DFS tak, aby v průběhu prohledávání vypisoval použité hrany (zprávy typu: *přecházím z uzlu  $u$  do uzlu  $v$* , každá hrana bude tedy vypsána dvakrát).
- (c) Modifikujte BFS aby vypsál nejkratší cestu do libovolného uzlu z uzlu  $s \in V$ .

### 3.3 Prohledávání do šířky

- (a) Na vstupu je graf  $G$  popisující plán sklepení. Uzly reprezentují křižovatky, hrany chodby. Některé chodby jsou uzamčené a na některých křižovatkách jsou umístěny klíče. Známe, který klíč patří ke které chodbě. Nalezněte nejkratší trasu, která nás dostane ze křižovatky  $s$  do křižovatky  $t$ .
1. Ve sklepení je pouze jediná zamčená chodba a jediný klíč.
  2. Ve sklepení je  $k$  zamčených chodeb a  $k$  klíčů,  $k \ll n$  (**0.5 bodu**).
- (b) Mějme graf  $G$  a funkci  $h : E \rightarrow \{0, 1\}$  ohodnocující hrany nulou či jedničkou. Nalezněte nejkratší cestu mezi uzly  $s$  a  $t$ , kde délku cesty počítáme jako součet hodnot použitých hran dle funkce  $h$ . (**0.5 bodu**)

## 3.4 Domácí úkol (0.5 b)

Mějme souvislý neorientovaný graf  $G = (V, E)$ . Chceme mazat vrcholy jeden po druhém tak, aby byl graf v průběhu mazání stále souvislý. Navrhněte algoritmus, který v čase  $\mathcal{O}(|V| + |E|)$  vypíše správné pořadí mazání vrcholů a dokažte jeho správnost.

*Můžete používat algoritmy z přednášky a známé datové struktury.  
Úkol odevzdejte na příštím cvičení (případně e-mailem).*