2.10 Silná souvislost

2.10.1 Silně souvislé grafy. Definice. Řekneme, že orientovaný graf G je *silně souvislý*, jestliže pro každou dvojici vrcholů u, v existuje orientovaná cesta z vrcholu u do vrcholu v a orientovaná cesta z vrcholu v do vrcholu u. □

2.10.2 Poznámka. V definici silně souvislého grafu jsme mohli požadovat pouze existenci orientované cesty z vrcholu u do vrcholu v. Je to proto, že existenci takové cesty vyžadujeme pro všechny dvojice vrcholů, tedy i pro dvojici v, u.

2.10.3 Tvrzení. Souvislý graf je silně souvislý právě tehdy, když každá hrana leží v nějakém cyklu. □

Zdůvodnění. Předpokládejme, že graf G je silně souvislý a vyberme jeho libovolnou hranu e s $PV(e) = x$, $KV(e) = y$. Protože je graf silně souvislý, existuje orientovaná cesta z vrcholu y do vrcholu x. Přidáme-li k této cestě hranu e, dostaneme cyklus, který e obsahuje.

Předpokládejme, že graf G je souvislý a každá jeho hrana je obsažena v některém cyklu. Vyberme libovolně dva vrcholu u, v v grafu. Protože je graf souvislý, existuje neorientovaná cesta z vrcholu u do vrcholu v. Každou hranu v této cestě, kterou jde cesta „proti směru hrany“, nahraďme částí cyklu, který tuto hranu obsahuje. Tím dostaneme orientovaný sled z u do v, a ten jistě obsahuje orientovanou cestu. Ukázali jsme, že G je silně souvislý.

2.10.4 Silně souvislé komponenty. Definice. Je dán orientovaný graf G. Množina vrcholů B se nazývá *silně souvislá komponenta*, též *komponenta silné souvislosti*, jestliže je maximální podmnožina vrcholů taková, podgraf indukovaný B je silně souvislý. □

Poznamenejme, že termín „maximální“ zde znamená „nedá se přidat vrchol při zachování silné souvislosti“. Jinými slovy, pro každý vrchol $v \notin B$ graf indukovaný množinou $B \cup \{v\}$ není silně souvislý.

2.10.5 Poznámka. Každý vrchol orientovaného grafu leží přesně v jedné silně souvislé komponentě. O hranách už takové tvrzení neplatí – mohou existovat hrany, které vedou mezi silně souvislými komponentami.

2.10.6 Hledání silně souvislých komponent — Kosaraju, Sharir algoritmus.

Vstup: Orientovaný graf G.
Výstup: Silně souvislé komponenty grafu G.

1. Prohledáme graf do hloubky a vypíšeme vrcholy v pořadí, ve kterém jsme vrcholy opouštěli.
2. V grafu G obrátíme hrany, dostaneme graf G'.
3. Prohledáme graf G' do hloubky a to v pořadí opačném pořadí v kroku 1.
4. Vrcholy stromů druhého prohledávání jsou pak vrcholy jednotlivých silně souvislých komponent.

2.10.7 Tarjanův algoritmus pro nalezení silně souvislých komponent. Existuje algoritmus, který nalezne silně souvislé komponenty a je rozšířením algoritmu pro prohledávání do hloubky DFS. Autorem algoritmu je Robert Tarjan.

Uvedeme myšlenku rozšíření DFS pro nalezení silně souvislých komponent. Kromě pořadových čísel budeme vrcholům přiřazovat tzv. zpětná čísla. Vrcholy budeme dělit do tří skupin:

- Ještě nenavštívené vrcholy, to budou vrcholy, které ještě nemají pořadové číslo ani zpětné číslo.

- Vrcholy, které již byly navštíveny, ale ještě nejsou zařazeny do žádné komponenty silné souvislosti.
- Vrcholy, které již byly zařazeny do některé komponenty silné souvislosti.

Vstup: orientovaný graf G.

Výstup: silně souvislé komponenty grafu G.

Pomocné proměnné: Každý vrchol x bude mít přiřazen dvě čísla – pořadové číslo $P(x)$ a zpětné číslo $Z(x)$. Číslo $P(x)$ udává pořadí, ve kterém byl vrchol x poprvé navštíven při prohledávání do hloubky; číslo $Z(x)$ udává nejmenší pořadové číslo vrcholu, který je z x orientovaně dostupný a to orientovanou cestou, která je tvořena několika hranami prohledávání a nejvýše jednou hranou zpět.

Dále používáme zásobník ZAS z prohledávání do hloubky.

1. *[Inicializace.]* Všechny vrcholy jsou nenavštívené a zásobník ZAS je prázdný.
2. *[Volba vrcholu].* Vybereme libovolný dosud nenavštívený vrchol x, označíme ho v, přiřadíme v jeho pořadové číslo $P(v)$, $Z(v) := P(v)$ a vložíme v na vrchol zásobníku ZAS.

   Jestliže nenavštívený vrchol neexistuje, výpočet končí.
3. *[Volba hrany e].* Vezmeme některou dosud nepoužitou hranu e začínající ve vrcholu v a označíme w koncový vrchol této hrany e.

   Pokud hrana neexistuje, pokračujeme krokem 7.
4. *[Koncový vrchol e nebyl navštíven.]* Je-li vrchol w ještě nenavštívený, přiřadíme mu pořadové číslo $P(w)$, položíme $Z(w) := P(w)$, zařadíme w na vrchol zásobníku ZAS a položíme $v := w$. Pokračujeme krokem 3.
5. *[Koncový vrchol e byl navštíven, ale není zařazen do některé komponenty.]* Je-li vrchol w ještě nezařazený do některé komponenty silné souvislosti, zkontrolujeme, zda $Z(v) > P(w)$. Jestliže ano, položíme $Z(v) := P(w)$ a pokračujeme krokem 3.
6. *[Koncový vrchol e je zařazen do některé komponenty.]* Je-li vrchol w již zařazen do některé komponenty silné souvislosti, pokračujeme krokem 3.
7. *[Možná nová komponenta.]* Jestliže $Z(v) = P(v)$, utvoříme novou komponentu silné souvislosti. Komponenta bude obsahovat dosud nezařazené vrcholy w, pro které platí $P(w) \geq P(v)$. Vrcholy komponenty odstraníme ze zásobníku, pokračujeme krokem 8.

   Je-li $Z(v) < P(v)$, pokračujeme krokem 8.
8. *[Test, zda je ZAS prázdný.]* Je-li zásobník ZAS prázdný, pokračujeme krokem 2.

   Není-li prázdný, označíme x vrchol zásobníku ZAS

   Jestliže nebyla uzavřena komponenta a $Z(x) > Z(v)$, položíme $Z(x) := Z(v)$.

   Dále položíme $v := x$ a pokračujeme krokem 3.

2.10.8 Poznámka. Oba algoritmy pracují v lineárním čase, to je v čase úměrném počtu hran a počtu vrcholů. Tarjanův algoritmus použije pouze jedno prohledávání do hloubky, algoritmus autorů Kosaraju a Sharir dvě prohledávání do hloubky. Poznamenejme, že Tarjanův algoritmus je součástí velmi rychlého (také lineárního) algoritmu pro zjištění, zda daný graf lze nakreslit do roviny tak, aby se jednotlivé hrany nekřížily (to je, zda je rovinný).

2.10.9 Kondenzace grafu. Definice. Je dán orientovaný graf $G = (V, E)$. *Kondenzace grafu* G je graf $\bar{G} = (\bar{V}, \bar{E})$, kde $\bar{V}$ je množina všech silně souvislých komponent grafu G a hrana vede z komponenty K_1 do komponenty K_2 právě tehdy, když $K_1 \neq K_2$ a existují vrcholy $u \in K_1$, $v \in K_2$ takové, že (u, v) je hrana grafu G. □

2.10.10 Poznámka. Kondenzace grafu už je vždy acyklický graf. Kdyby totiž nebyl, pak byly špatně spočítané silně souvislé komponenty.