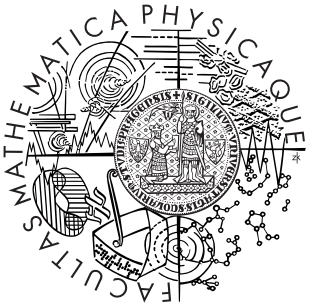
Algoritmy a datové struktury 1

2/2 Z+Zk, NTIN060

pomocné slajdy k datovým strukturám

Pavel Veselý (IUUK)



vesely+ads1@iuuk.mff.cuni.cz

https://iuuk.mff.cuni.cz/∼vesely/vyuka/LS2122/ads1.html

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:
- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:
- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)
- prioritní fronta — implementace haldou

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:

- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)
- prioritní fronta — implementace haldou
- problém Union-Find (udržování komponent souvislosti grafu)

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:

- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)
- prioritní fronta — implementace haldou
- problém Union-Find (udržování komponent souvislosti grafu)
- reprezentace množiny $X \subseteq \mathcal{U}$, kde je $\mathcal{U}$ je univerzum

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:

- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)
- prioritní fronta — implementace haldou
- problém Union-Find (udržování komponent souvislosti grafu)
- reprezentace množiny $X \subseteq \mathcal{U}$, kde je $\mathcal{U}$ je univerzum
- slovník — množina dvojic (k, v), kde k je klíč a v hodnota

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:

- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)
- prioritní fronta — implementace haldou
- problém Union-Find (udržování komponent souvislosti grafu)
- reprezentace množiny $X \subseteq \mathcal{U}$, kde je $\mathcal{U}$ je univerzum
- slovník — množina dvojic (k, v), kde k je klíč a v hodnota
- uspořádaná množina / slovník

Datové struktury

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy: **1.** „černá skříňka s daným rozhraním“

2. konkrétní implementace

Příklady:
- fronta (*first in, first out*, FIFO)
- zásobník (*last in, first out*, LIFO)
- prioritní fronta — implementace haldou
- problém Union-Find (udržování komponent souvislosti grafu)
- reprezentace množiny $X \subseteq \mathcal{U}$, kde je $\mathcal{U}$ je univerzum
- slovník — množina dvojic (k, v), kde k je klíč a v hodnota
- uspořádaná množina / slovník

Příklady implementací:
- pole, spojový seznam, binární halda
- binární vyhledávací strom
- hešovací tabulka

Binární vyhledávací stromy (BVS)

Binary search tree (BST)

- pro problém uspořádaného slovníku (nebo uspořádané množiny klíčů)

1. AVL stromy = hloubkově vyvážené stromy

- G. M. **A**deľson-**V**elskij a Je. M. **L**andis '62
- pro každý vrchol se hloubka podstromů levého a pravého syna liší max. o 1
- mají logaritmickou hloubku

Binární vyhledávací stromy (BVS)

Binary search tree (BST)

- pro problém uspořádaného slovníku (nebo uspořádané množiny klíčů)

1. AVL stromy = hloubkově vyvážené stromy

- G. M. **A**deľson-**V**elskij a Je. M. **L**andis '62
- pro každý vrchol se hloubka podstromů levého a pravého syna liší max. o 1
- mají logaritmickou hloubku

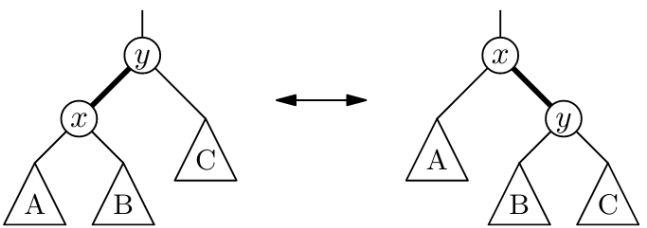
Binární vyhledávací stromy (BVS)

Binary search tree (BST)

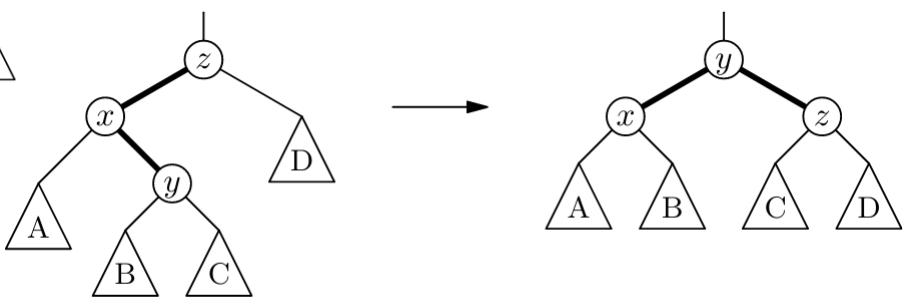
- pro problém uspořádaného slovníku (nebo uspořádané množiny klíčů)

1. AVL stromy = hloubkově vyvážené stromy

- G. M. **A**deľson-**V**elskij a Je. M. **L**andis '62
- pro každý vrchol se hloubka podstromů levého a pravého syna liší max. o 1
- mají logaritmickou hloubku



Obrázek 8.5: Jednoduchá rotace

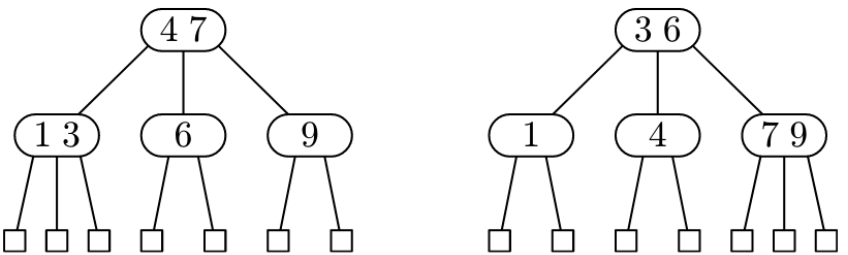


Obrázek 8.6: Dvojitá rotace

Vícecestné vyhledávací stromy

(a, b)-**stromy** pro $a \geq 2, b \geq 2a - 1$

- Každý vnitřní vrchol má a až b synů, kořen má 2 až b synů
- Listy na stejné hladině a virtuální (`NULL`)

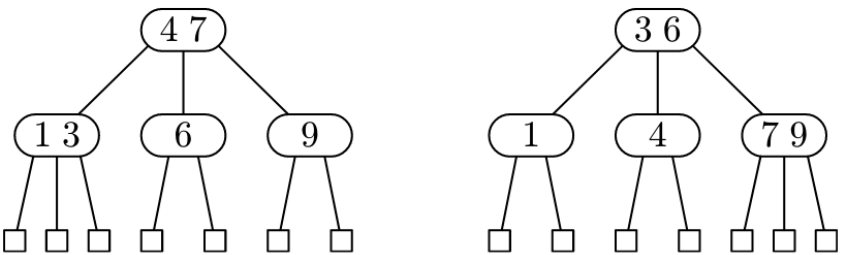


Obrázek 8.7: Dva (2, 3)-stromy pro tutéž množinu klíčů

Vícecestné vyhledávací stromy

(a, b)-stromy pro $a \geq 2, b \geq 2a - 1$

- Každý vnitřní vrchol má a až b synů, kořen má 2 až b synů
- Listy na stejné hladině a virtuální (`NULL`)



Obrázek 8.7: Dva (2, 3)-stromy pro tutéž množinu klíčů

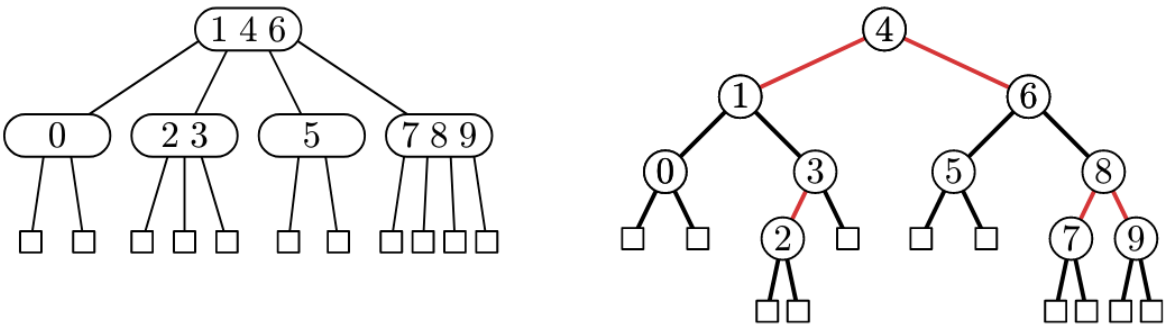
B-stromy [Bayer, McCreight '70]

- $(a, 2a - 1)$- nebo $(a, 2a)$-stromy
- data někdy jen nejnižší hladině nad listy, vyšší hladiny obsahují pomocné klíče

Červeno-černé stromy [Bayer '72]

varianta left-leaning (LLRB) [Sedgewick '08]

- překlad $(2, 4)$-stromů na binární stromy

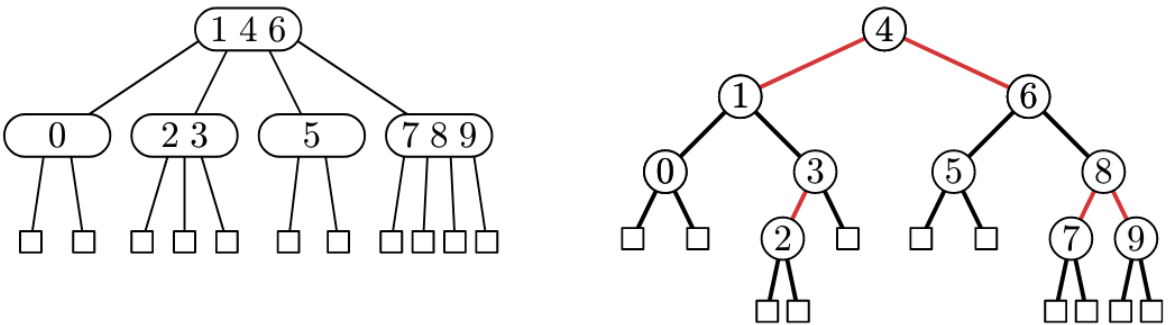


Obrázek 8.11: Překlad $(2, 4)$-stromu na LLRB strom

Červeno-černé stromy [Bayer '72]

varianta left-leaning (LLRB) [Sedgewick '08]

- překlad $(2, 4)$-stromů na binární stromy



Obrázek 8.11: Překlad $(2, 4)$-stromu na LLRB strom

Jiná varianta **červeno-černých** stromů:

- Každý vrchol je **černý** nebo **červený**
- Kořen a (virtuální) listy jsou **černé**
- Pokud je vrchol **červený**, oba jeho synové jsou **černí**
- Pro lib. vrchol v platí, že každá cesta z v do listu má stejný # **černých** vrcholů.

Binární vyhledávací stromy: shrnutí

- AVL stromy: menší hloubka, náročnější udržovat
- (a, b)-stromy: vhodné pro více úrovní paměti, popř. databáze
- Červeno-černé stromy: lze dostat $O(1)$ rotací při mazání

Binární vyhledávací stromy: shrnutí

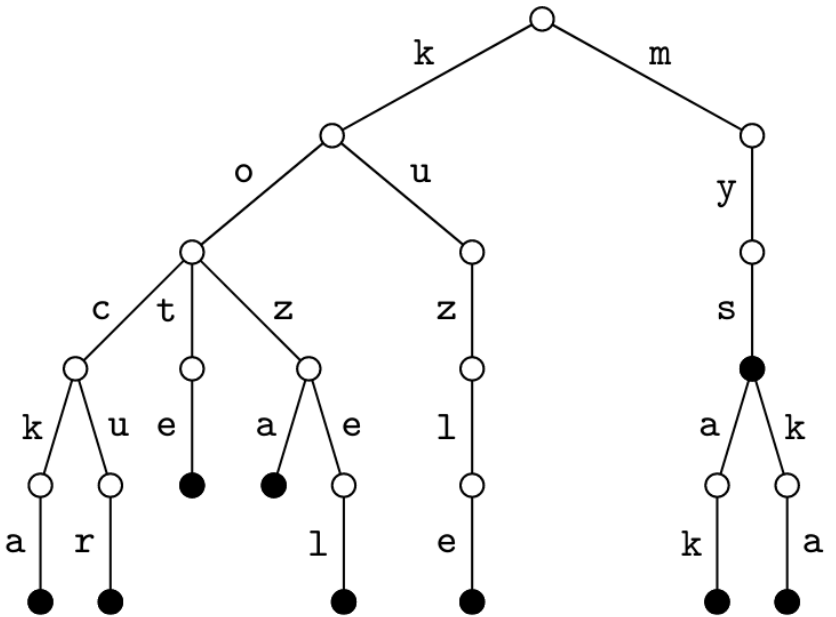
- AVL stromy: menší hloubka, náročnější udržovat
- (a, b)-stromy: vhodné pro více úrovní paměti, popř. databáze
- Červeno-černé stromy: lze dostat $O(1)$ rotací při mazání

Splay stromy

- nevyvážený BVS
- při přístupu na vrchol vyrotujeme vrchol do kořene
- preferujeme dvojité rotace
- amortizovaná složitost $O(\log n)$

Písmenkový strom (trie, prefixový strom)

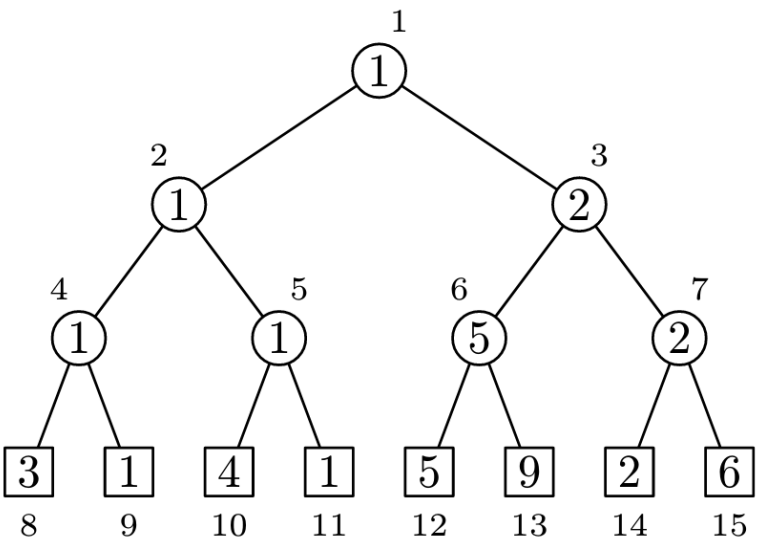
- trie: ze slov trie a retrieval



Obrázek 4.4: Písmenkový strom pro slova `kocka`, `kocur`, `kote`, `koza`, `kozel`, `kuzle`, `mys`, `mysak`, `myska`

Intervalový strom

- dána posloupnost čísel $x_0, \ldots, x_{n-1}$
- zodpovídá intervalové dotazy na minimum / maximum / součet / ...
- umí zpracovat aktualizaci hodnot posloupnosti
  - včetně intervalového updatu (líně)



Obrázek 4.6: Intervalový strom a jeho očíslování

Hešování (Hašování, Hashování) — **Hashing**

- pro reprezentaci množiny / slovníku bez uspořádání
  - pokud nepotřebujeme hledat k-tý nejmenší klíč, součet hodnot klíčů v intervalu, apod.
- „náhodnou“ funkcí h si zobrazíme množinu $X \subseteq \mathcal{U}$ do hešovací tabulky
  - hešovací tabulka $\sim$ pole velikosti $m = \Theta(n)$, kde $n = |X|$
  - samotné univerzum $\mathcal{U}$ může být obrovské

Hešování (Hašování, Hashování) — **Hashing**

- pro reprezentaci množiny / slovníku bez uspořádání
  - pokud nepotřebujeme hledat k-tý nejmenší klíč, součet hodnot klíčů v intervalu, apod.
- „náhodnou“ funkcí h si zobrazíme množinu $X \subseteq \mathcal{U}$ do hešovací tabulky
  - hešovací tabulka $\sim$ pole velikosti $m = \Theta(n)$, kde $n = |X|$
  - samotné univerzum $\mathcal{U}$ může být obrovské
- kolize = dva prvky z X skončí ve stejné buňce

Hešování (Hašování, Hashování) — Hashing

- pro reprezentaci množiny / slovníku bez uspořádání
  - pokud nepotřebujeme hledat k-tý nejmenší klíč, součet hodnot klíčů v intervalu, apod.
- „náhodnou“ funkcí h si zobrazíme množinu $X \subseteq \mathcal{U}$ do hešovací tabulky
  - hešovací tabulka $\sim$ pole velikosti $m = \Theta(n)$, kde $n = |X|$
  - samotné univerzum $\mathcal{U}$ může být obrovské
- kolize = dva prvky z X skončí ve stejné buňce

Řešení kolizí:

1. Separované řetízky — spojový seznam v každé buňce
   - průměrná délka řetízku je n/m, stejně tak střední hodnota
   - hledání, přidávání a mazání *průměrně* v čase $O(1 + n/m)$

Hešování (Hašování, Hashování) — Hashing

- pro reprezentaci množiny / slovníku bez uspořádání
  - pokud nepotřebujeme hledat k-tý nejmenší klíč, součet hodnot klíčů v intervalu, apod.
- „náhodnou“ funkcí h si zobrazíme množinu $X \subseteq \mathcal{U}$ do hešovací tabulky
  - hešovací tabulka $\sim$ pole velikosti $m = \Theta(n)$, kde $n = |X|$
  - samotné univerzum $\mathcal{U}$ může být obrovské
- kolize = dva prvky z X skončí ve stejné buňce

Řešení kolizí:

1. Separované řetízky — spojový seznam v každé buňce
   - průměrná délka řetízku je n/m, stejně tak střední hodnota
   - hledání, přidávání a mazání *průměrně* v čase $O(1 + n/m)$
2. Otevřená adresace — v buňce tabulky max. jeden klíč

Hešování (Hašování, Hashování) — **Hashing**

- pro reprezentaci množiny / slovníku bez uspořádání
  - pokud nepotřebujeme hledat k-tý nejmenší klíč, součet hodnot klíčů v intervalu, apod.
- „náhodnou“ funkcí h si zobrazíme množinu $X \subseteq \mathcal{U}$ do hešovací tabulky
  - hešovací tabulka $\sim$ pole velikosti $m = \Theta(n)$, kde $n = |X|$
  - samotné univerzum $\mathcal{U}$ může být obrovské
- kolize = dva prvky z X skončí ve stejné buňce

Řešení kolizí:

1. Separované řetízky — spojový seznam v každé buňce
   - průměrná délka řetízku je n/m, stejně tak střední hodnota
   - hledání, přidávání a mazání *průměrně* v čase $O(1 + n/m)$
2. Otevřená adresace — v buňce tabulky max. jeden klíč

Přehešovávání: pokud n/m příliš vzroste, zvětšíme m dvakrát

- amortizovaná složitost $O(1)$ na vložení prvku

Hešování a narozeniny

Nechť h je úplně náhodná (každý prvek z $\mathcal{U}$ zahešován uniformně a nezávisle na ostatních).

Jak velké m zvolit, aby nenastala kolize s pravděpodobností třeba 99 %?

Hešování a narozeniny

Nechť h je úplně náhodná (každý prvek z $\mathcal{U}$ zahešován uniformně a nezávisle na ostatních).

Jak velké m zvolit, aby nenastala kolize s pravděpodobností třeba 99 %?

- narozeninový paradox
  - pokud je v místnosti $\geq$ 23 lidí, s pravděpodobností $\geq$ 50 % jsou tu dva se stejnými narozeninami

Hešování a narozeniny

Nechť h je úplně náhodná (každý prvek z $\mathcal{U}$ zahešován uniformně a nezávisle na ostatních).

Jak velké m zvolit, aby nenastala kolize s pravděpodobností třeba 99 %?

- narozeninový paradox
  - pokud je v místnosti $\geq$ 23 lidí, s pravděpodobností $\geq$ 50 % jsou tu dva se stejnými narozeninami
- potřebovali bychom $m \in \Omega(n^2)$ pro konstantní pravděpodobnost žádné kolize

Univerzální a k-nezávislé hešování

Definice. Systém funkcí $\mathcal{H}$ z univerza $\mathcal{U}$ do $[m] = \{0, \ldots, m-1\}$ je c-univerzální pro $c \geq 1$, pokud $\forall x, y \in \mathcal{U}, x \neq y : \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq \frac{c}{m}$.

- př. 1-univerzálního systému: $\mathcal{H}_{\text{lin}} = \{h_{a,b} \mid a, b \in [p], a \neq 0\}$
  - kde $h_{a,b} = ((ax + b) \mod p) \mod m$ a $p \geq |\mathcal{U}|$ je prvočíslo

Univerzální a k-nezávislé hešování

Definice. Systém funkcí $\mathcal{H}$ z univerza $\mathcal{U}$ do $[m] = \{0, \ldots, m-1\}$ je c-univerzální pro $c \geq 1$, pokud $\forall x, y \in \mathcal{U}, x \neq y : \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq \frac{c}{m}$.

- př. 1-univerzálního systému: $\mathcal{H}_{\text{lin}} = \{h_{a,b} \mid a, b \in [p], a \neq 0\}$
  - kde $h_{a,b} = ((ax + b) \mod p) \mod m$ a $p \geq |\mathcal{U}|$ je prvočíslo

---

Definice. Systém funkcí $\mathcal{H}$ z univerza $\mathcal{U}$ do $[m]$ je $(2, c)$-nezávislý pro $c \geq 1$, pokud

$$\forall x_1, x_2 \in \mathcal{U}, x_1 \neq x_2, \forall y_1, y_2 \in [m] : \Pr_{h \in \mathcal{H}}[h(x_1) = y_1 \wedge h(x_2) = y_2] \leq \frac{c}{m^2}.$$

- př. $(2, c)$-nezávislého systému: $\mathcal{H}_{\text{lin}}$
- někdy se $(2, c)$-nezávislosti říká silná c-univerzalita

Univerzální a k-nezávislé hešování

Definice. Systém funkcí $\mathcal{H}$ z univerza $\mathcal{U}$ do $[m] = \{0, \ldots, m-1\}$ je c-univerzální pro $c \geq 1$, pokud $\forall x, y \in \mathcal{U}, x \neq y : \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq \frac{c}{m}$.

- př. 1-univerzálního systému: $\mathcal{H}_{\text{lin}} = \{h_{a,b} \mid a, b \in [p], a \neq 0\}$
  - kde $h_{a,b} = ((ax + b) \mod p) \mod m$ a $p \geq |\mathcal{U}|$ je prvočíslo

---

Definice. Systém funkcí $\mathcal{H}$ z univerza $\mathcal{U}$ do $[m]$ je $(2, c)$-nezávislý pro $c \geq 1$, pokud

$$\forall x_1, x_2 \in \mathcal{U}, x_1 \neq x_2, \forall y_1, y_2 \in [m] : \Pr_{h \in \mathcal{H}}[h(x_1) = y_1 \wedge h(x_2) = y_2] \leq \frac{c}{m^2}.$$

- př. $(2, c)$-nezávislého systému: $\mathcal{H}_{\text{lin}}$
- někdy se $(2, c)$-nezávislosti říká silná c-univerzalita

Definice. Systém funkcí $\mathcal{H}$ z univerza $\mathcal{U}$ do $[m]$ je k-nezávislý pro $k \geq 2$, pokud

$$\forall x_1, \ldots, x_k \in \mathcal{U} \textbf{ různé}, \forall y_1, \ldots, y_k \in [m] : \Pr_{h \in \mathcal{H}}[h(x_1) = y_1 \wedge \cdots \wedge h(x_k) = y_k] \lesssim \frac{1}{m^k}.$$

- př.: funkce $\left(\left(\sum_{i=0}^{k-1} a_i x^i\right) \mod p\right) \mod m$ a $p \geq |\mathcal{U}|$ je prvočíslo