

6 Vlastnosti funkcí a Skládání relací

Vraťme se nyní k látce Lekce 3. Z jejího pokročilého obsahu jsme doposud velmi detailně probírali relace a jejich jednotlivé vlastnosti. Nyní se podívejme, jak lze relace mezi sebou „skládat“, což je například základní technika práce s relačními databázemi.

datum	let	letadlo
5.11.	OK535	ČSA
5.11.	AF2378	ČSA
5.11.	DL5457	ČSA
6.11.	OK535	AirFrance
6.11.	AF2378	AirFrance

○

pasažér	datum	let
Petr	5.11.	OK535
Pavel	6.11.	OK535
Jan	5.11.	AF2378
Josef	5.11.	DL5457
Alena	6.11.	AF2378

=

pasažér	letadlo
Petr	ČSA
Josef	ČSA
...	...

□

Je však i jiné místo, kde jste se zajisté se skládáním relací setkali – jedná se o skládání funkcí. Jak například spočítáte na kalkulačce výsledek složitějšího vzorce? □

Stručný přehled lekce

- * Přehled základních vlastností funkcí.
- * Inverzní relace a skládání (binárních) relací, příklady.
- * Skládání funkcí (coby relací), speciálně aplikováno na permutace.
- * Induktivní definice množin a funkcí.

Zopakování relací a funkcí

- *Relace* mezi množinami A_1, \dots, A_k , pro $k \in \mathbb{N}$, je **libovolná** podmnožina kartézského součinu

$$R \subseteq A_1 \times \dots \times A_k.$$

Pokud $A_1 = \dots = A_k = A$, hovoříme o *k-ární relaci* R na A . \square

- (*Totální*) *funkce* z množiny A do množiny B je *relace* f mezi A a B taková, že pro každé $x \in A$ existuje **právě jedno** $y \in B$ takové, že $(x, y) \in f$. \square
 - * Množina A se nazývá *definiční obor* a množina B *obor hodnot* funkce f . Funkcím se také říká *zobrazení*.
 - * Místo $(x, y) \in f$ píšeme obvykle $f(x) = y$. Zápís

$$f : A \rightarrow B$$

říká, že f je funkce s def. oborem A a oborem hodnot B . \square

- Pokud naši definici funkce upravíme tak, že požadujeme pro každé $x \in A$ **nejvýše jedno** $y \in B$ takové, že $(x, y) \in f$, obdržíme definici *parciální funkce* z A do B .

6.1 Vlastnosti funkcí

Definice: Funkce $f : A \rightarrow B$ je

- **injektivní** (nebo také **prostá**) právě když pro každé $x, y \in A$, $x \neq y$ platí, že $f(x) \neq f(y)$; \square
- **surjektivní** (nebo také „na“) právě když pro každé $y \in B$ existuje $x \in A$ takové, že $f(x) = y$; \square
- **bijektivní** (vzáj. **jednoznačná**) právě když je injektivní a souč. surjektivní. \square

Ukázky vlastností funkcí.

- Funkce **plus** : $\mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ je surjektivní, ale není prostá. \square
- Funkce $g : \mathbb{Z} \rightarrow \mathbb{N}$ daná předpisem

$$g(x) = \begin{cases} -2x - 1 & \text{jestliže } x < 0, \\ 2x & \text{jinak} \end{cases}$$

je bijektivní. \square

- Funkce $\emptyset : \emptyset \rightarrow \emptyset$ je bijektivní. \square
- Funkce $\emptyset : \emptyset \rightarrow \{a, b\}$ je injektivní, ale není surjektivní.

6.2 Inverzní relace a skládání relací

Definice: Necht' $R \subseteq A \times B$ je **binární** relace mezi A a B . **Inverzní relace k relaci** R se značí R^{-1} a je definována takto:

$$R^{-1} = \{(b, a) \mid (a, b) \in R\}$$

(R^{-1} je tedy relace **mezi B a A** .) \square

Příklady inverzí pro relace–funkce.

- Inverzí **bijektivní** funkce $f(x) = x + 1$ na \mathbb{Z} je funkce $f^{-1}(x) = x - 1$. \square
- Inverzí **prosté** funkce $f(x) = e^x$ na \mathbb{R} je **parciální** funkce $f^{-1}(x) = \ln x$. \square
- Funkce $g(x) = x \bmod 3$ **není prostá** na \mathbb{N} , a proto její inverzí je „**jen**“ relace $g^{-1} = \{(a, b) \mid a = b \bmod 3\}$.
Konkrétně $g^{-1} = \{(0, 0), (0, 3), (0, 6), \dots, (1, 1), (1, 4), \dots, (2, 2), (2, 5), \dots\}$. \square

Tvrzení 6.1. Mějme funkci $f : A \rightarrow B$. Pak její inverzní relace f^{-1} je

- a) **parciální funkce** právě když f je **prostá**, \square
- b) **funkce** právě když f je **bijektivní**.

Důkaz vyplývá přímo z definic funkce a inverze relace. \square

Definice 6.2. Složení (kompozice) relací R a S .

Nechť $R \subseteq A \times B$ a $S \subseteq B \times C$ jsou binární relace. **Složení** relací R a S (v **tomto pořadí!**) je relace $S \circ R \subseteq A \times C$ definovaná takto:

$$S \circ R = \{(a, c) \mid \text{existuje } b \in B \text{ takové, že } (a, b) \in R, (b, c) \in S\} \square$$

Složení relací čteme „ R složeno s S “ nebo (pozor na pořadí!) „ S po R “. \square

Příklady skládání relací.

- Je-li

$$* A = \{a, b\}, \quad B = \{1, 2\}, \quad C = \{X\},$$

$$* R = \{(a, 1), (b, 1), (b, 2)\}, \quad S = \{(1, X)\},$$

pak složením vznikne relace

$$* S \circ R = \{(a, X), (b, X)\}. \quad \square$$

- Složením funkcí $h(x) = x^2$ a $f(x) = x + 1$ na \mathbb{R} vznikne funkce

$$f \circ h(x) = f(h(x)) = x^2 + 1. \quad \square$$

- Složením těchto funkcí „**naopak**“ ale vznikne funkce $h \circ f(x) = h(f(x)) = (x+1)^2$.

Poznámka: Nepříjemné je, že v některých oblastech matematiky (například v algebře při skládání zobrazení) se setkáme s **právě opačným** zápisem skládání, kdy se místo $S \circ R$ píše $R \cdot S$ nebo jen RS .

6.3 Skládání relací „v praxi“

Příklad 6.3. Skládání v relační databázi studentů, jejich předmětů a fakult.

Mějme dvě binární relace – jednu R přiřazující studentům MU kódy jejich zapsaných předmětů, druhou S přiřazující kódy předmětů jejich mateřským fakultám.

R :

student (učo)	předmět (kód)
121334	MA010
133935	M4135
133935	IA102
155878	M1050
155878	IB000

S :

předmět (kód)	fakulta MU
MA010	FI
IB000	FI
IA102	FI
M1050	PřF
M4135	PřF

Jak z těchto „tabulkových“ relací zjistíme, kteří studenti mají **zapsané předměty na kterých fakultách** (třeba na FI)? □

Jedná se jednoduše o **složení relací** $S \circ R$. V našem příkladě třeba:

$S \circ R$:

student (učo)	fakulta MU
121334	FI
133935	FI
133935	PřF
155878	FI
155878	PřF

Zobecněné skládání relací

Fakt (skládání relací vyšší arity): Mějme relace $T \subseteq K_1 \times K_2 \times \dots \times K_k$ a $U \subseteq L_1 \times L_2 \times \dots \times L_\ell$, přičemž pro nějaké $m < \min(k, \ell)$ platí $L_1 = K_{k-m+1}, L_2 = K_{k-m+2}, \dots, L_m = K_k$. Pak relaci T lze složit s relací U na zvolených m složkách L_1, \dots, L_m („překrytí“) s použitím Definice 6.2 takto:

- Položme $A = K_1 \times \dots \times K_{k-m}$, $B = L_1 \times \dots \times L_m$ a $C = L_{m+1} \times \dots \times L_\ell$.
- Příslušné relace pak jsou
 - * $R = \{(\vec{a}, \vec{b}) \in A \times B \mid (a_1, \dots, a_{k-m}, b_1, \dots, b_m) \in T\}$ a
 - * $S = \{(\vec{b}, \vec{c}) \in B \times C \mid (b_1, \dots, b_m, c_{m+1}, \dots, c_\ell) \in U\}$. □
- Nakonec přirozeně položíme $U \circ_m T \simeq S \circ R$, takže vyjde $U \circ_m T = \{(\vec{a}, \vec{c}) \mid \text{ex. } \vec{b} \in B, \text{ že } (a_1, \dots, a_{k-m}, b_1, \dots, b_m) \in T \text{ a } (b_1, \dots, b_m, c_{m+1}, \dots, c_\ell) \in U\}$.

Schematicky pro snazší orientaci:

$$\begin{array}{l} T \subseteq \boxed{K_1 \times \dots \times K_{k-m}} \times \boxed{K_{k-m+1} \times \dots \times K_k} \\ U \subseteq \underbrace{\quad \quad \quad}_{L_1 \times \dots \times L_m} \times L_{m+1} \times \dots \times L_\ell \\ U \circ_m T \subseteq \underbrace{\boxed{K_1 \times \dots \times K_{k-m}}}_A \times \underbrace{\quad \quad \quad}_B \times \underbrace{\boxed{L_{m+1} \times \dots \times L_\ell}}_C \end{array}$$

Příklad 6.4. Skládání v relační databázi pasažérů a letů u leteckých společností.

Podívejme se na příklad hypotetické rezervace letů pro cestující, relace T . Jak známo (tzv. codeshare), letecké společnosti si mezi sebou „dělí“ místa v letadlech, takže různé lety (podle kódů) jsou ve skutečnosti realizovány stejným letadlem jedné ze společností. To zase ukazuje relace U .

T :

pasažér	datum	let
Petr	5.11.	OK535
Pavel	6.11.	OK535
Jan	5.11.	AF2378
Josef	5.11.	DL5457
Alena	6.11.	AF2378

U :

datum	let	letadlo
5.11.	OK535	ČSA
5.11.	AF2378	ČSA
5.11.	DL5457	ČSA
6.11.	OK535	AirFrance
6.11.	AF2378	AirFrance

Ptáme-li se nyní, setkají se Petr a Josef na palubě stejného letadla? Případně, či letadlo to bude? Odpovědi nám dá zase složení relací $U \circ_2 T$, jak je posáno výše.

$U \circ_2 T$:

pasažér	letadlo
Petr	ČSA
Josef	ČSA
...	...

Zkuste se zamyslet, lze tyto dvě relace skládat ještě jinak? Co by pak bylo významem? □

6.4 Skládání funkcí, permutace

Fakt: Mějme zobrazení (funkce) $f : A \rightarrow B$ a $g : B \rightarrow C$. Pak jejich **složení** $(g \circ f) : A \rightarrow C$ definované

$$(g \circ f)(x) = g(f(x)). \square$$

- Jak například na běžné kalkulačce vypočteme hodnotu funkce $\sin^2 x$? \square
Složíme (v tomto pořadí) „elementární“ funkce $f(x) = \sin x$ a $g(x) = x^2$. \square
- Jak bychom na „elementární“ funkce rozložili aritmetický výraz $2 \log(x^2 + 1)$? \square
Ve správném pořadí složíme funkce $f_1(x) = x^2$, $f_2(x) = x + 1$, $f_3(x) = \log x$ a $f_4(x) = 2x$. \square
- A jak bychom obdobně vyjádřili složením funkcí aritmetický výraz $\sin x + \cos x$?
Opět je odpověď přímočará, vezmeme „elementární“ funkce $g_1(x) = \sin x$ a $g_2(x) = \cos x$, a pak je „složíme“ další funkcí $h(x, y) = x + y$. \square
Vidíme však, že takto pojaté „skládání“ už nezapadá hladce do našeho formalismu skládání relací.

Skládání permutací

Definice: Necht' *permutace* π množiny $\{1, 2, \dots, n\}$ je určena seřazením jejích prvků (p_1, \dots, p_n) . Pak π je zároveň **bijektivním zobrazením** $\{1, \dots, n\} \rightarrow \{1, \dots, n\}$ definovaným předpisem $\pi(i) = p_i$. \square

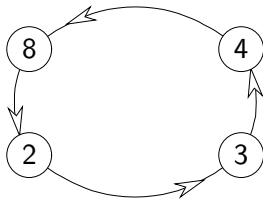
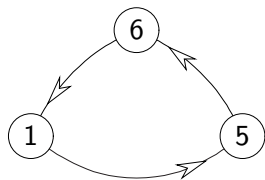
Tudíž lze permutace *skládat jako relace* podle Definice 6.2. \square

Poznámka: Všechny permutace množiny $\{1, 2, \dots, n\}$ spolu s operací skládání tvoří grupu, zvanou symetrická grupa S_n .

Permutační grupy (podgrupy symetrické grupy) jsou velice důležité v algebře, neboť každá grupa je vlastně isomorfní některé permutační grupě. \square

Příkladem permutace vyskytující se v programátorské praxi je třeba zobrazení $i \mapsto (i+1) \bmod n$ ("inkrement"). \square Často se třeba lze setkat (aniž si to mnohdy uvědomujeme) s permutacemi při indexaci prvků polí.

V kontextu pohledu na funkce a jejich skládání coby relací si zavedeme jiný, názornější, způsob zápisu permutací – pomocí jejich **cyklů**.



Definice: Necht' π je permutace na množině A . **Cyklem** v π rozumíme posloupnost $\langle a_1, a_2, \dots, a_k \rangle$ různých prvků A takovou, že $\pi(a_i) = a_{i+1}$ pro $i = 1, 2, \dots, k - 1$ a $\pi(a_k) = a_1$. \square

Jak název napovídá, v zápise cyklu $\langle a_1, a_2, \dots, a_k \rangle$ není důležité, kterým prvkem začneme, ale jen dodržení cyklického pořadí. Cyklus v permutaci může mít i jen jeden prvek (zobrazený na sebe).

Například permutace $(5, 3, 4, 8, 6, 1, 7, 2)$ je zakreslena svými cykly výše.

Věta 6.5. Každou permutaci π na konečné množině A lze zapsat jako složení cyklů na disjunktních podmnožinách (rozkladu) A . \square

Důkaz: Vezmeme libovolný prvek $a_1 \in A$ a iterujeme zobrazení $a_2 = \pi(a_1)$, $a_3 = \pi(a_2)$, atd., až se dostaneme „zpět“ k $a_{k+1} = \pi(a_k) = a_1$. Proč tento proces skončí? Protože A je konečná a tudíž ke zopakování některého prvku a_{k+1} musí dojít. Nadto je π prostá, a proto nemůže nastat $\pi(a_k) = a_j$ pro $j > 1$. Takto získáme první cyklus $\langle a_1, \dots, a_k \rangle$. \square

Induktivně pokračujeme s hledáním dalších cyklů ve zbylé množině $A \setminus \{a_1, \dots, a_k\}$, dokud nezůstane prázdná. \square

Značení permutací cykly: Necht' se permutace π podle Věty 6.5 skládá z cyklů $\langle a_1, \dots, a_k \rangle$, $\langle b_1, \dots, b_l \rangle$ až třeba $\langle z_1, \dots, z_m \rangle$. Pak zapíšeme

$$\pi = (\langle a_1, \dots, a_k \rangle \langle b_1, \dots, b_l \rangle \dots \langle z_1, \dots, z_m \rangle) \square.$$

Primitivní pseudonáhodné generátory v počítačích iterují z náhodného počátku permutací danou vztahem $i \mapsto (i + p) \bmod q$. Je pochopitelné, že tato permutace nesmí obsahovat krátké cykly, lépe řečeno, měla by se skládat z jediného (dlouhého) cyklu.

Příklad 6.6. Ukázka skládání permutací daných svými cykly.

Vezměme 7-prvkovou permutaci $(5, 3, 4, 2, 6, 1, 7)$. Ta se rozkládá na tři cykly $\langle 1, 5, 6 \rangle$, $\langle 2, 3, 4 \rangle$ a $\langle 7 \rangle$. Jiná permutace $(3, 4, 5, 6, 7, 1, 2)$ se skládá z jediného cyklu $\langle 1, 3, 5, 7, 2, 4, 6 \rangle$. \square

Nyní určíme složení těchto dvou permutací (zápisem cykly):

$$(\langle 1, 5, 6 \rangle \langle 2, 3, 4 \rangle \langle 7 \rangle) \circ (\langle 1, 3, 5, 7, 2, 4, 6 \rangle) = (\langle 1, 4 \rangle \langle 2 \rangle \langle 3, 6, 5, 7 \rangle)$$

(Nezapomínejme, že první se ve složení aplikuje pravá permutace!) \square

Postup skládání jsme použili následovný:

- 1 se zobrazí v permutaci vpravo na 3 a pak vlevo na 4. \square
- Následně 4 se zobrazí na 6 a pak na 1. Tím „uzavřeme“ první cyklus $\langle 1, 4 \rangle$. \square
- Dále se 2 zobrazí na 4 a pak hned zpět na 2, tj. má samostatný cyklus. \square
- Zbylý cyklus $\langle 3, 6, 5, 7 \rangle$ určíme analogicky. \square

6.5 Induktivní definice množin a funkcí

Přímým zobecněním rekurentních definic je následující koncept.

Definice 6.7. Induktivní definice množiny.

Jedná se obecně o popis (nějaké) množiny M v následujícím tvaru:

- Je dáno několik pevných (*bázických*) prvků $a_1, a_2, \dots, a_k \in M$. \square
- Je dán soubor *induktivních pravidel* typu

Jsou-li (libovolné prvky) $x_1, \dots, x_\ell \in M$, pak také $y \in M$.

V tomto případě je y typicky funkcí $y = f_i(x_1, \dots, x_\ell)$. \square

Pak naše *induktivně definovaná množina* M je určena jako nejmenší (inkluzí) množina vyhovující těmto pravidlům.

Vidíte podobnost této definice s uzávěrem relace? (Věta 5.8.) \square

Pro nejbližší příklad induktivní definice se obrátíme na množinu všech přirozených čísel.

- $0 \in \mathbb{N}$
- Je-li $i \in \mathbb{N}$, pak také $i + 1 \in \mathbb{N}$.

Pro každé $y \in \mathbb{N}$ můžeme definovat jinou množinu $M_y \subseteq \mathbb{N}$ induktivně takto:

- $y \in M_y$.
- Jestliže $x \in M_y$ a $x + 1$ je liché, pak $x + 2 \in M_y$.

Pak například $M_3 = \{3\}$, nebo $M_4 = \{4 + 2i \mid i \in \mathbb{N}\}$. \square

Definice: Řekneme, že daná induktivní definice množiny M je *jednoznačná*, právě když každý prvek M lze odvodit z bazových prvků pomocí induktivních pravidel právě *jedním způsobem*. \square

Definujme množinu $M \subseteq \mathbb{N}$ induktivně takto:

- $2, 3 \in M$.
- Jestliže $x, y \in M$ a $x \leq y$, pak také $x^2 + y^2$ a $x \cdot y$ jsou prvky M .

Proč tato induktivní definice není jednoznačná? \square Například číslo $8 \in M$ lze odvodit způsobem $8 = 2 \cdot (2 \cdot 2)$, ale zároveň zcela jinak $8 = 2^2 + 2^2$.

V čem tedy spočívá důležitost jednoznačných induktivních definic množin?

Definice 6.8. Induktivní definice funkce z induktivní množiny.

Nechť množina M je dána **jednoznačnou** induktivní definicí. Pak říkáme, že funkce $\mathcal{F} : M \rightarrow X$ je definována **induktivně** (vzhledem k induktivní definici M), pokud je řečeno:

- Pro každý z bázeckých prvků $a_1, a_2, \dots, a_k \in M$ je určeno $\mathcal{F}(a_i) = c_i$. \square
- Pro každé induktivní pravidlo typu

“Jsou-li (libovolné prvky) $x_1, \dots, x_\ell \in M$, pak také $f(x_1, \dots, x_\ell) \in M$ ”

je definováno

$\mathcal{F}(f(x_1, \dots, x_\ell))$ na základě hodnot $\mathcal{F}(x_1), \dots, \mathcal{F}(x_\ell)$. \square

Pro příklad se podívejme třeba do manuálu unixového příkazu `test` **EXPRESSION**:

`EXPRESSION` is true or false and sets exit status. It is one of:

<code>! EXPRESSION</code>	<code>EXPRESSION</code> is false
<code>EXPRESSION1 -a EXPRESSION2</code>	both <code>EXPRESSION1</code> and <code>EXPRESSION2</code> are true
<code>EXPRESSION1 -o EXPRESSION2</code>	either <code>EXPRESSION1</code> or <code>EXPRESSION2</code> is true
<code>[-n] STRING</code>	the length of <code>STRING</code> is nonzero
<code>STRING1 = STRING2</code>	the strings are equal

.....

Induktivní definice se „strukturální“ indukci

Příklad 6.9. Jednoduché aritmetické výrazy

Nechť (abeceda) $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, \odot, \oplus, (,)\}$. Definujme množinu *jednoduchých výrazů* $SExp \subseteq \Sigma^*$ induktivně takto:

- Dekadický zápis každého přirozeného čísla n je prvek $SExp$.
- Jestliže $x, y \in SExp$, pak také $(x) \odot (y)$ a $(x) \oplus (y)$ jsou prvky $SExp$. \square

(Jak vidíme, díky „závorkování“ je tato induktivní definice **jednoznačná**.)

Pro „vyhodnocení“ výrazu pak definujme funkci $Val : SExp \rightarrow \mathbb{N}$ **induktivně**:

- Bázické prvky: $Val(n) = n$, kde n je dekadický zápis přirozeného čísla n . \square
- První induktivní pravidlo: $Val((x) \oplus (y)) = Val(x) + Val(y)$.
- Druhé induktivní pravidlo: $Val((x) \odot (y)) = Val(x) \cdot Val(y)$. \square

(Tímto způsobem jsme našim výrazům vlastně přiřadili jejich „význam“, sémantiku.) \square

Příklad 6.10. Důkaz správnosti přiřazeného „významu“ $Val : SExp \rightarrow \mathbb{N}$.

Věta. Pro každý výraz $s \in SExp$ je hodnota $Val(s)$ číselně rovna výsledku vyhodnocení výrazu s podle běžných zvyklostí aritmetiky. \square

Matematickou **indukcí**: Na rozdíl od dříve probíraných příkladů zde nevidíme žádný celočíselný „parametr n “, a proto si jej budeme muset nejprve definovat. \square

Naši indukci tedy povedeme podle „délky ℓ odvození výrazu s “ definované jako **počet aplikací induktivních pravidel** potřebných k odvození $s \in SExp$.

Takto aplikované matematické indukci se často říká **strukturální indukce**. \square

Důkaz: V **bázi indukce** ověříme vyhodnocení bázeckých prvků, což jsou zde dekadické zápisy přirozených čísel. Platí $Val(n) = n$, což skutečně odpovídá zvyklostem aritmetiky. \square

V **indukčním kroku** se podíváme na vyhodnocení $Val((x) \oplus (y)) = Val(x) + Val(y)$. Podle běžných zvyklostí aritmetiky by hodnota $Val((x) \oplus (y))$ měla být rovna **součtu** vyhodnocení **výrazu** x , což je podle indukčního předpokladu rovno $Val(x)$ (x má zřejmě kratší délku odvození), a vyhodnocení **výrazu** y , což je podle indukčního předpokladu rovno $Val(y)$. Takže skutečně $Val((x) \oplus (y)) = Val(x) + Val(y)$. Druhé pravidlo $Val((x) \odot (y))$ se dořeší analogicky. \square