PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

BAKALÁŘSKÁ PRÁCE

Interaktivní program na podporu výuky binárních relací

2010

Martin Sommer

Anotace

Cílem práce bylo prostudovat téma binární relace. Poté vytvořit interaktivní program, který by byl užitečný komukoli, kdo by se rád seznámil se základními pojmy, které souvisí s binárními relacemi. Program je rozdělen na výukovou a praktickou část. Výuková část poskytuje uživateli informace k tématu binárních relací. Praktická část obsahuje interaktivní příklady, které slouží k lepšímu pochopení těchto informací a na kterých si uživatel může ověřit své znalosti.

Děkuji mému vedoucímu práce RNDr. Miroslavu Kolaříkovi, Ph.D., za trpělivost a podnětné připomínky k mé práci a také za umožnění volnosti při realizaci.

Obsah

Seznam obrázků

1. Teoretické základy

1.1. Pojem relace a binární relace

Relace je matematickým protějškem pojmu vztah. Např. číslo 4 je ve vztahu „být větší“ s číslem 2, nebo množina $\{1, 2\}$ je ve vztahu „být podmnožinou“ s množinou $\{1, 2, 3\}$ a není ve vztahu „být podmnožinou“ s množinou $\{2\}$. Každý vztah je určen tzv. aritou a množinami, jejichž prvky do vztahu vstupují. Arita udává počet prvků ve vztahu. Vztah, jehož arita je dva nazveme binární.

Příklad 1. *Máme množinu X všech žáků nějaké školy, množinu Y všech učitelů téže školy. Můžeme uvažovat vztah „x je žákem y“, kde x je některý žák školy a y některý učitel školy. Takový vztah je binární a množiny, jejichž prvky do vztahu vstupují jsou X a Y.*

Nyní budeme definovat pojmy související s relací, viz [1].

Definice 1. *Uspořádaná n-tice prvků $x_1, \ldots, x_n$ (v tomto pořadí) se označuje $\langle x_1, \ldots, x_n \rangle$. Prvek x_i $(1 \leq i \leq n)$ se nazývá i-tá složka n-tice. Dvě uspořádané n-tice jsou si rovny, právě když mají stejný počet složek a odpovídající si složky jsou stejné.*

Definice 2. *Kartézský součin množin $X_1, \ldots, X_n$ je množina $X_1 \times \cdots \times X_n = \{\langle x_1, \ldots, x_n \rangle \mid x_1 \in X_1, \ldots, x_n \in X_n\}$.*

Definice 3. *Relace mezi množinami $X_1, \ldots, X_n$ je libovolná podmnožina kartézského součinu $X_1 \times \cdots \times X_n$.*

Relace je tedy množina sestávající z n-tic prvků příslušných množin. Číslo n je arita relace a danou relaci nazýváme n-ární.

Příklad 2. *Nechť je množina $A = \{a, b, c\}$, množina $B = \{1, 2\}$, množina $C = \{Jirka, Petr\}$, kartézský součin $A \times B \times C = \{\langle a, 1, Jirka\rangle, \langle a, 1, Petr\rangle, \langle a, 2, Jirka\rangle, \langle a, 2, Petr\rangle, \langle b, 1, Jirka\rangle, \langle b, 1, Petr\rangle, \langle b, 2, Jirka\rangle, \langle b, 2, Petr\rangle, \langle c, 1, Jirka\rangle, \langle c, 1, Petr\rangle, \langle c, 2, Jirka\rangle, \langle c, 2, Petr\rangle\}$, kartézský součin $A \times B = \{\langle a, 1\rangle, \langle a, 2\rangle, \langle b, 1\rangle, \langle b, 2\rangle, \langle c, 1\rangle, \langle c, 2\rangle\}$. Pak ternární relace $R = \{\langle a, 1, Jirka\rangle, \langle b, 2, Jirka\rangle, \langle b, 2, Petr\rangle, \langle c, 2, Jirka\rangle\}$, $R \subseteq A \times B \times C$ a binární relace $L = \{\langle a, 1\rangle, \langle b, 1\rangle, \langle c, 2\rangle, \langle a, 2\rangle\}$, $L \subseteq A \times B$.*

Binární relace je tedy libovolná podmnožina kartézského součinu dvou libovolných množin. Pokud jsou si dvě množiny (mezi kterými relaci uvažujeme) rovny, pak mluvíme o binární relaci na množině. Pokud jsou různé, pak mluvíme o binární relaci mezi množinami.

Mezi speciální relace na nějaké množině X patří:

- prázdná relace $\emptyset$
- relace identity $\omega_X = \{\langle x, x\rangle \mid x \in X\}$
- kartézský čtverec $\iota_X = X \times X$.

1.2. Operace s binárními relacemi

Relace jsou definovány jako množiny a proto s nimi lze provádět množinové operace (např. $\cap, \cup, -$).
Máme-li množiny X, Y a relace R, L, kde $R, L \subseteq X \times Y$, potom relace:

- $R \cap L = \{x \mid x \in R \text{ a } x \in L\}$
- $R \cup L = \{x \mid x \in R \text{ nebo } x \in L\}$
- $R - L = \{x \mid x \in R \text{ a } x \notin L\}$.

Příklad 3. *Nechť je dána množina* $X = \{1, 2, 3\}$, *R a L jsou relace na množině* X, $R = \{\langle 1, 2\rangle, \langle 1, 3\rangle, \langle 2, 3\rangle\}$, $L = \{\langle 1, 2\rangle, \langle 2, 2\rangle, \langle 2, 3\rangle\}$. *Pak* $R \cap L = \{\langle 1, 2\rangle, \langle 2, 3\rangle\}$, $R \cup L = \{\langle 1, 2\rangle, \langle 1, 3\rangle, \langle 2, 2\rangle, \langle 2, 3\rangle\}$, $R - L = \{\langle 1, 3\rangle\}$.

Pro binární relace můžeme definovat (díky jejich struktuře) další operace, které nejsou obdobami operací s množinami. Mezi ně patří inverzní relace a skládání relací.

Inverzní relace k nějaké relaci R se značí R^{-1} a je definovaná předpisem $R^{-1} = \{\langle y, x\rangle \mid \langle x, y\rangle \in R\}$. Tedy relaci R^{-1} získáme záměnou složek v uspořádaných dvojicích relace R.

Příklad 4. *Nechť je množina* $X = \{a, b, c\}$, *R je relace na množině* X, $R = \{\langle a, b\rangle, \langle b, c\rangle, \langle c, a\rangle\}$. *Pak inverzní relace k R je* $R^{-1} = \{\langle b, a\rangle, \langle c, b\rangle, \langle a, c\rangle\}$.

Nyní k operaci skládání. Je-li R relace mezi množinami X, Y a L relace mezi množinami Y, Z, potom složení R a L je relace $R \circ L$ mezi množinami X, Z definovaná předpisem:

$$R \circ L = \{\langle x, z\rangle \mid \text{existuje } y \in Y : \langle x, y\rangle \in R \text{ a } \langle y, z\rangle \in L\}.$$

Příklad 5. *Nechť je množina* $X = \{a, b, c\}$, *množina* $Y = \{1, 2, 3\}$, *množina* $Z = \{Jirka, Petr, Martin\}$, *R je relace mezi množinami X a Y*, $R = \{\langle a, 1\rangle, \langle a, 3\rangle, \langle c, 2\rangle\}$, *L je relace mezi množinami Y a Z*, $L = \{\langle 3, Petr\rangle, \langle 3, Martin\rangle, \langle 2, Jirka\rangle\}$. *Pak* $R \circ L$ *je relace mezi množinami X a Z*, $R \circ L = \{\langle a, Petr\rangle, \langle a, Martin\rangle, \langle c, Jirka\rangle\}$.

1.3. Reprezentace binárních relací

Existují různé způsoby, jak reprezentovat relaci. Nyní si některé ukážeme a popíšeme jejich výhody i nevýhody.

1.3.1. Reprezentace pomocí matice

Nechť máme množiny A, B. Pokud matice reprezentuje relaci $R \subseteq A \times B$, potom řádky (resp. sloupce) představují prvky množiny A (resp. B). Je-li prvek $x \in A$ v relaci R s prvkem $y \in B$ potom hodnota matice na průsečíku řádku, který představuje prvek x, a sloupce, který představuje prvek y, bude 1. V opačném případě 0.

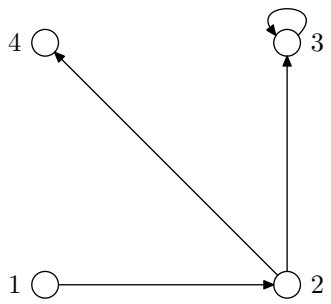
Příklad 6. *Nechť je množina $A = \{a, b, c\}$, množina $B = \{1, 2, 3\}$, R je relace mezi množinami A a B, $R = \{\langle a, 1\rangle, \langle a, 2\rangle, \langle b, 2\rangle, \langle c, 1\rangle\}$. Pak S je matice reprezentující relaci R, $S = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$.*

Přehlednost maticové reprezentace relace závisí na mohutnosti množin, mezi kterými relaci uvažujeme. Zjistit, zda dva prvky jsou spolu v relaci je snadné a rychlé. Ovšem nevýhodou může být její velikost. Např. matice reprezentující nějakou relaci R na nějaké množině X, která obsahuje 100 prvků, je typu 100×100 a obsahuje tedy 10000 hodnot. Využití hodnot matice je závislé na počtu uspořádaných dvojic, které leží v relaci R. Např. pokud R obsahuje 10 prvků, pak matice bude obsahovat 10 jedniček a zbytek budou nuly. Přitom uchovávat informaci o nulách je zbytečné. Srovnej s [1].

1.3.2. Reprezentace pomocí grafu

Relaci R na množině X můžeme znázornit pomocí grafu, přičemž každý prvek $x \in X$ znázorníme jako uzel grafu a jestliže $\langle x, y\rangle \in R$, pak nakreslíme orientovanou hranu od uzlu představující prvek x k uzlu představující prvek y.

Příklad 7. *Nechť je množina $X = \{1, 2, 3, 4\}$, R je relace na množině X, $R = \{\langle 1, 2\rangle, \langle 2, 3\rangle, \langle 2, 4\rangle, \langle 3, 3\rangle\}$, pak graf relace R je znázorněn na obrázku 1.*

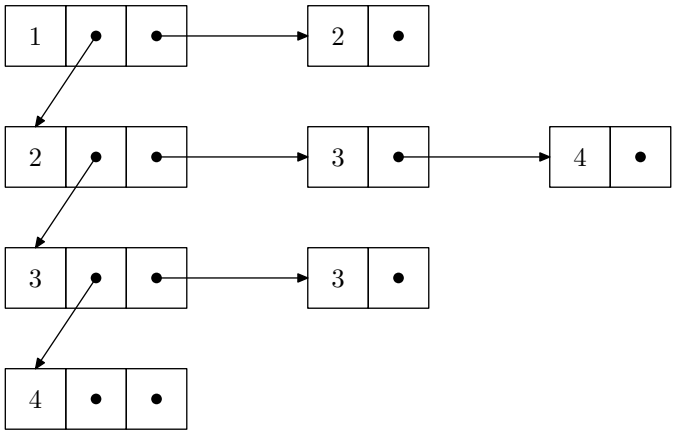


Obrázek 1. Graf binární relace

Grafová reprezentace je přehledná a především názorná. Ale s rostoucím počtem uzlů a vztahů mezi nimi, může ztrácet na přehlednosti.

1.3.3. Reprezentace seznamem seznamů

Pokud chceme využívat relaci na počítači, musíme ji vhodně reprezentovat v paměti počítače. Reprezentace seznamem seznamů je vhodná pro zpracování počítačem a narozdíl od maticové reprezentace je paměťově úsporná pro relace s malou mohutností na množině s velkou mohutností. Na obrázku 2 je znázorněna reprezentace relace R z předchozího příkladu pomocí seznamu seznamů.



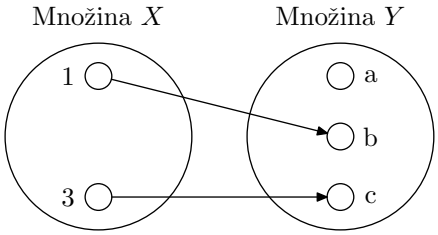
Obrázek 2. Reprezentace relace pomocí seznamu seznamů.

1.4. Zobrazení a relace

Zobrazení lze považovat za zvláštní případ relace. Od pojmu relace se dostaneme k pojmu zobrazení, máme-li relaci B mezi dvěma množinami X, Y, kde každý prvek z X je v relaci B právě s jedním prvkem z Y. Taková relace odpovídá zobrazení X do Y a můžeme ho značit $\beta : X \to Y$. Čili při zobrazení β se každý prvek z množiny X zobrazí na právě jeden prvek z množiny Y. Prvek $y \in Y$, který odpovídá prvku $x \in X$ při zobrazení β se značí $\beta(x)$ a nazývá se obraz prvku x. Každý prvek $x \in X$, jehož obrazem při zobrazení β je prvek $y \in Y$, se nazývá vzor prvku y. Každý prvek z množiny X při zobrazení β má právě jeden obraz, ale prvek z množiny Y nemusí mít žádný vzor, nebo může mít jeden či více vzorů. Množina všech vzorů prvku $y \in Y$ se nazývá úplný vzor. Srovnej s [7]. Zobrazení $\beta : X \to Y$ je:

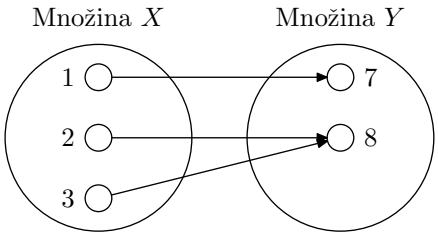
- injektivní, pokud ke každému prvku $y \in Y$ existuje nejvýše jeden vzor
- surjektivní, pokud každý prvek $y \in Y$ má alespoň jeden vzor
- bijektivní, pokud je injektivní a surjektivní.

Příklad 8. *Nechť je množina* $X = \{1, 3\}$*, množina* $Y = \{a, b, c\}$*, pak zobrazení* $\beta : X \to Y$*,* $\beta(1) = b$*,* $\beta(3) = c$*, znázorněné na obrázku 3 je injektivní a není surjektivní.*



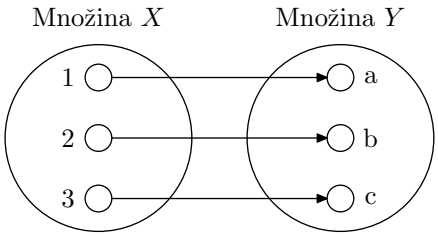
Obrázek 3. Injektivní zobrazení

Příklad 9. *Nechť je množina* $X = \{1, 2, 3\}$*, množina* $Y = \{7, 8\}$*, pak zobrazení* $\beta : X \to Y$*,* $\beta(1) = 7$*,* $\beta(2) = 8$*,* $\beta(3) = 8$*, znázorněné na obrázku 4 je surjektivní a není injektivní.*



Obrázek 4. Surjektivní zobrazení

Příklad 10. *Nechť je množina* $X = \{1, 2, 3\}$*, množina* $Y = \{a, b, c\}$*, pak zobrazení* $\beta : X \to Y$*,* $\beta(1) = a$*,* $\beta(2) = b$*,* $\beta(3) = c$*, znázorněné na obrázku 5 je bijektivní.*



Obrázek 5. Bijektivní zobrazení

1.5. Vlastnosti binárních relací na množině

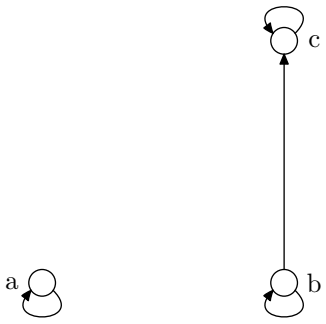
Binární relace na množině může mít některé vlastnosti, které nám později budou sloužit k vyčlenění určitých tříd relací.

Definice 4. *Máme-li množinu A a relaci $R \subseteq A \times A$, pak R je:*

- *reflexivní, pokud pro každé $x \in A$ platí $\langle x, x\rangle \in R$*
- *symetrická, pokud pro každé $\langle x, y\rangle \in R$ platí $\langle y, x\rangle \in R$*
- *antisymetrická, pokud pro každé $\langle x, y\rangle \in R$ platí: je-li ($\langle x, y\rangle \in R$ a $\langle y, x\rangle \in R$) pak $x = y$*
- *úplná, pokud pro každé $x, y \in A$ platí $\langle x, y\rangle \in R$ nebo $\langle y, x\rangle \in R$*
- *tranzitivní, pokud pro každé $x, y, z \in R$ platí: je-li ($\langle x, z\rangle \in R$ a $\langle z, y\rangle \in R$) pak $\langle x, y\rangle \in R$*

Tedy relace R je reflexivní, jestliže každý prvek z A je v relaci R se sebou samým. Graf reflexivní relace R bude mít u každého prvku smyčku a matice samé jedničky na hlavní diagonále.

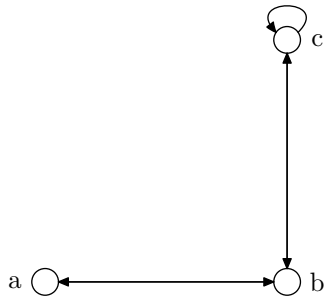
Příklad 11. *Nechť je množina $A = \{a, b, c\}$, R je reflexivní relace na množině A, $R = \{\langle a, a\rangle, \langle b, c\rangle, \langle b, b\rangle, \langle c, c\rangle\}$, pak $S = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$ je matice relace R a graf relace R je znázorněn na obrázku 6.*



Obrázek 6. Graf reflexivní relace

Relace R je symetrická, když pro každou dvojici $\langle x, y\rangle$ ležící v R platí, že také $\langle y, x\rangle$ leží v R. V grafu symetrické relace R mezi prvky $x, y \in A$ není buď žádná hrana, nebo vede hrana z x do y i z y do x. Matice symetrické relace R je vždy symetrická podle hlavní diagonály.

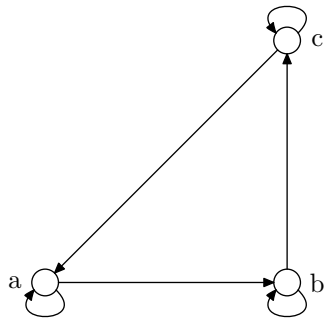
Příklad 12. *Nechť je množina* $A = \{a, b, c\}$*,* R *je symetrická relace na množině* A*,* $R = \{\langle a, b\rangle, \langle b, c\rangle, \langle b, a\rangle, \langle c, b\rangle, \langle c, c\rangle\}$*, pak* $S = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$ *je matice relace* R *a graf relace* R *je znázorněn na obrázku 7.*



Obrázek 7. Graf symetrické relace

Relace R je úplná, jestliže každé dva prvky z množiny A jsou porovnatelné. Každá dvě pole matice R, která jsou souměrná podle hlavní diagonály, obsahují alespoň jednu jedničku. Mezi každými dvěma uzly grafu relace R je alespoň jedna orientovaná hrana.

Příklad 13. *Nechť je množina* $A = \{a, b, c\}$*,* R *je úplná relace na množině* A*,* $R = \{\langle a, b\rangle, \langle c, a\rangle, \langle b, c\rangle, \langle a, a\rangle, \langle b, b\rangle, \langle c, c\rangle\}$*, pak* $S = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$ *je matice relace* R *a graf relace* R *je znázorněn na obrázku 8.*

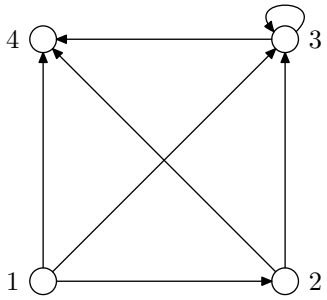


Obrázek 8. Graf úplné relace

Relace R je tranzitivní, jestliže je $x \in A$ v relaci R s $y \in A$ a zároveň y v relaci R se $z \in A$, potom je x v relaci R se z. Graf relace R se vyznačuje tím, že jestliže

se z uzlu x do uzlu z dostaneme po dvou orientovaných hranách přes uzel y, pak z x do z můžeme „jít“ přímo.

Příklad 14. *Nechť je množina* $A = \{1, 2, 3, 4\}$*,* R *je tranzitivní relace na množině* A*,* $R = \{\langle 1,2\rangle, \langle 2,3\rangle, \langle 1,3\rangle, \langle 3,3\rangle, \langle 3,4\rangle, \langle 2,4\rangle, \langle 1,4\rangle\}$*, pak* $S = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ *je matice relace* R *a graf relace* R *je znázorněn na obrázku 9.*



Obrázek 9. Graf tranzitivní relace

Dále se zmíníme o tzv. uzávěrech relací, viz [3].
Řekneme, že vlastnost V binárních relací je uzavíratelná, jestliže splňuje tyto podmínky:

- pro každou množinu A a relaci $R \subseteq A \times A$ mající vlastnost V, existuje relace $S \subseteq A \times A$ mající vlastnost V a $R \subseteq S$.
- průnik všech relací na množině A, které mají vlastnost V, je relace mající vlastnost V.

Tedy reflexivita, symetrie a tranzitivita i jejich libovolné kombinace jsou uzavíratelné vlastnosti. Např. antisymetrie není.

Nechť V je uzavíratelná vlastnost binárních relací, máme-li libovolnou relaci R na množině A a množinu M všech relací S na množině A, kde $R \subseteq S$, majících vlastnost V, potom V-uzávěr relace R je nejmenší (vzhledem k množinové inkluzi) relace z množiny M. Tedy:

- reflexivní uzávěr R je $R \cup \{\langle x, x\rangle \mid x \in A\}$
- symetrický uzávěr R je $R \cup \{\langle x, y\rangle \mid \langle y, x\rangle \in R\}$
- tranzitivní uzávěr R je $\bigcup_{n=1}^{\infty} R^n$.

Pokud je množina A konečná, pak můžeme pozměnit definici tranzitivního uzávěru relace R na přijatelnější tvar. Je-li $|A| \leq n$ a $\langle x, y\rangle \in R^{n+1}$, pak $\langle x, y\rangle \in R^m$ pro nějaké $m \in \mathbf{N}$ splňující $m \leq n$. Tedy pro každé $k \in \mathbf{N}$, kde $k > n = |A|$ platí, že $R^k \subseteq \bigcup_{i=1}^{n} R^i$. Takže tranzitivní uzávěr relace R na množině $|A| = n$, je relace $\bigcup_{i=1}^{n} R^i$.

1.6. Relace uspořádání

Nejprve si ukážeme několik příkladů uspořádání.

Příklad 15. *Uspořádání na množině všech celých čísel. Pro každá dvě různá celá čísla umíme určit, které je větší než druhé. Všechna čísla jsou vzájemně porovnatelná a můžeme je uspořádat podle velikosti. Dané uspořádání nemůže být symetrické, protože je-li např.* $1 \leq 2$*, pak nemůže platit* $2 \leq 1$*. Platí zde reflexivita, protože každý prvek z této množiny je „menší nebo rovno“ sám se sebou. Máme-li tři celá čísla a první je „menší nebo rovno“ jak druhé, druhé je „menší nebo rovno“ jak třetí, potom první je „menší nebo rovno“ jak třetí. Tedy uspořádání na množině celých čísel je tranzitivní.*

Příklad 16. *Máme-li množinu* $M = \{1, 2, 3\}$ *a* 2^M *množinu všech podmnožin množiny* M*, potom prvky potenční množiny* 2^M *můžeme uspořádat pomocí inkluze* $\subseteq$*. Ovšem neplatí, narozdíl od předešlého příkladu, že každé dva prvky jsou porovnatelné. Např.* $\{1\} \not\subseteq \{2, 3\}$ *a* $\{2, 3\} \not\subseteq \{1\}$*. Takovéto uspořádání je reflexivní, antisymetrické a tranzitivní.*

Definice 5. *Binární relaci na množině, která je reflexivní, antisymetrická a tranzitivní nazveme uspořádání. Úplné uspořádání se nazývá lineární uspořádání nebo řetězec. Máme-li relaci uspořádání* R *na množině* A*, pak* $\langle A, R\rangle$ *nazveme uspořádanou množinou.*

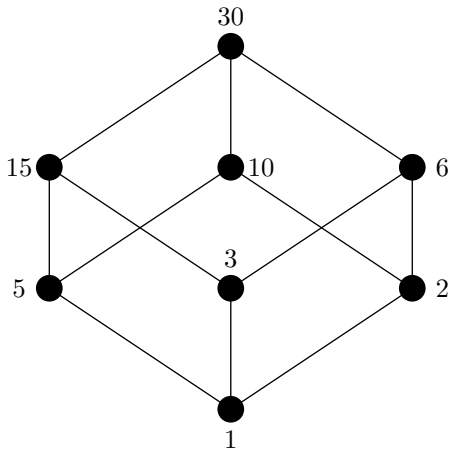
Protože uspořádání je relace, tak ji můžeme reprezentovat pomocí matice nebo grafu. Ovšem vhodnější způsob reprezentace je pomocí tzv. Hasseových diagramů. Při konstrukci Hasseova diagramu dané relace uspořádání využíváme tzv. relaci pokrytí. Máme-li relaci uspořádání R na množině A, pak relace pokrytí příslušná R je:

$$P = \{\langle x, y\rangle \mid \langle x, y\rangle \in R \text{ a } x \neq y \text{ a pro každé } z \in A \text{ platí: pokud } \langle x, z\rangle \in R \text{ a } \langle z, y\rangle \in R, \text{ pak } z \in \{x, y\}\}.$$

Hasseův diagram relace R se skládá z uzlů reprezentujících prvky množiny A a hran, které vyznačují relaci pokrytí P příslušné R. Pokud $\langle x, y\rangle \in P$, pak uzel reprezentující prvek x bude ležet níže než uzel y. Srovnej s [2].

Příklad 17. *Nechť je množina* $A = \{1, 2, 3, 5, 6, 10, 15, 30\}$*,* R *je relace uspořádání podle dělitelnosti na množině* A*,* $R = \{\langle 1, 1\rangle, \langle 1, 2\rangle, \langle 1, 3\rangle, \langle 1, 5\rangle, \langle 1, 6\rangle,$

$\langle 1,10\rangle$, $\langle 1,15\rangle$, $\langle 1,30\rangle$, $\langle 2,2\rangle$, $\langle 2,6\rangle$, $\langle 2,10\rangle$, $\langle 2,30\rangle$, $\langle 3,3\rangle$, $\langle 3,6\rangle$, $\langle 3,15\rangle$, $\langle 3,30\rangle$, $\langle 5,5\rangle$, $\langle 5,10\rangle$, $\langle 5,15\rangle$, $\langle 5,30\rangle$, $\langle 6,6\rangle$, $\langle 6,30\rangle$, $\langle 10,10\rangle$, $\langle 10,30\rangle$, $\langle 15,15\rangle$, $\langle 15,30\rangle$, $\langle 30,30\rangle\}$, *pak relace pokrytí příslušná* R *je* $P = \{\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,5\rangle, \langle 2,6\rangle, \langle 2,10\rangle, \langle 3,6\rangle, \langle 3,15\rangle, \langle 5,10\rangle, \langle 5,15\rangle, \langle 6,30\rangle, \langle 10,30\rangle, \langle 15,30\rangle\}$ *a příslušný Hasseův diagram relace* R *je znázorněn na obrázku 10.*



Obrázek 10. Hasseův diagram relace uspořádání

Některé prvky množiny $\langle A, R\rangle$ z předchozího příkladu vykazují určité vlastnosti. Neexistuje žádný prvek množiny A, který by dělil prvek 1, kromě něho samotného. Navíc prvek 1 dělí všechny prvky množiny A. Dále žádný prvek z množiny A není dělitelný prvkem 30, kromě něho samotného. Ale každý prvek z množiny A dělí prvek 30. Prvek 1 je dle následující definice minimální a nejmenší vzhledem k R a prvek 30 je maximální a největší vzhledem k R.

Dále můžeme vzít libovolnou podmnožinu A_1 množiny A a zjišťovat zda existují prvky z A, které jsou dělitelné všemi prvky z A_1, nebo dělí všechny prvky z A_1. Např. prvky 3, 6 dělí prvky 6, 30 a jsou dělitelné prvky 1, 3.

Definice 6. *Prvek* $x \in \langle A, R\rangle$ *se nazývá:*

- *minimální, jestliže pro každé* $y \in A$ *platí: pokud* $\langle y, x\rangle \in R$, *pak* $x = y$
- *nejmenší, jestliže pro každé* $y \in A$ *platí:* $\langle x, y\rangle \in R$
- *maximální, jestliže pro každé* $y \in A$ *platí: pokud* $\langle x, y\rangle \in R$, *pak* $x = y$
- *největší, jestliže pro každé* $y \in A$ *platí:* $\langle y, x\rangle \in R$.

Definice 7. *Máme-li uspořádanou množinu* $\langle A, R\rangle$ *a* $Y \subseteq A$ *potom množina:*

- $L(Y) = \{x \in A \mid$ *pro každé* $y \in Y$ *platí* $\langle x, y\rangle \in R\}$ *se nazývá dolní kužel množiny* Y *vzhledem k* $\langle A, R\rangle$

- $U(Y) = \{x \in A \mid$ *pro každé* $y \in Y$ *platí* $\langle y, x\rangle \in R\}$ *se nazývá horní kužel množiny* Y *vzhledem k* $\langle A, R\rangle$

Největší prvek $L(Y)$, pokud existuje, nazveme infimum a nejmenší prvek $U(Y)$ nazveme supremum.

1.7. Rozklady a relace ekvivalence

Relace ekvivalence je matematickým protějškem pojmu stejnost (totožnost, shodnost, rovnost, zaměnitelnost). Určení, zda nějaké objekty jsou stejné, závisí buď na situaci, v níž dané objekty uvažujeme, nebo závisí na pozorovateli, který z jeho hlediska posuzuje stejnost těchto objektů. Např. mějme šachovou soupravu. Bílé pěšce můžeme považovat vzhledem ke všem bílým figurám za zaměnitelné. Ovšem uvažujeme-li bílé pěšce v průběhu hry vzhledem k jejich pozicím, pak nemusí být zaměnitelní. Dále dáme-li bílé pěšce chlapci, který si chce hrát na vojáčky, potom z jeho pohledu může být každý pěšec jiný.

Máme množinu M objektů, z nichž některé jsou vzájemně zaměnitelné. M_x je množina všech objektů vzájemně zaměnitelných s objektem x. Zřejmě $x \in M_x$ a $M = \bigcup_{x\in M} M_x$. Předpokládejme, že $M_x \cap M_y \neq \emptyset$. Tedy existuje prvek z, který patří do množiny M_x i M_y. Prvek x je zaměnitelný s prvkem z, který je zaměnitelný s prvkem y. Tedy prvek x je zaměnitelný s prvkem y a také s každým prvkem množiny M_y. Proto $M_y \subseteq M_x$. Podobnou úvahou zjístíme, že platí $M_x \subseteq M_y$. Čili množiny vyskytující se ve sjednocení $M = \bigcup_{x\in M} M_x$ jsou si buď rovny, nebo jsou disjunktní. Viz [7].

Definice 8. *Systém neprázdných podmnožin množiny* M *značíme* $\mathcal{M}$ *a nazýváme rozklad množiny* M*, jestliže platí:*

- *sjednocení všech množin systému* $\mathcal{M}$ *se rovná množině* M
- *jestliže* $M_i, M_j \in \mathcal{M}$ *a* $M_i \cap M_j \neq \emptyset$*, pak* $M_i = M_j$

Množiny systému $\mathcal{M}$ *nazveme třídy rozkladu* $\mathcal{M}$*.*

Definice 9. *Relace* A *na množině* M *se nazývá relace ekvivalence (nebo jen ekvivalence) na množině* M*, jestliže existuje takový rozklad* $\mathcal{M}$ *množiny* M*, že* $\langle x, y\rangle \in A$*, právě když prvky* x *a* y *patří do stejné třídy rozkladu* $\mathcal{M}$*. Prvky* x*,* y *nazveme ekvivalentní.*

Z předchozí definice mimo jiné vyplývá, že ke každé ekvivalenci A na množině M existuje rozklad $\mathcal{M}$ na množině M. Systém $\mathcal{M}$ nazveme rozklad na množině M příslušný ekvivalenci A. Je známo, že ekvivalence a rozklady jsou ve vzájemně jednoznačné korespondenci. Tedy ke každému rozkladu na množině existuje ekvivalence na této množině. Je-li $\mathcal{M}$ nějaký rozklad množiny M, pak existuje relace A na množině M, kde $\langle x, y\rangle \in A$, právě když prvky x a y patří do stejné třídy M_i rozkladu $\mathcal{M}$. Relace A je ekvivalence na M a nazveme ji ekvivalence příslušná rozkladu $\mathcal{M}$. Pro takovouto A platí:

- pro každé $x \in M$ platí $\langle x, x \rangle \in A$ (prvek x patří do stejné třídy rozkladu $\mathcal{M}$ jako prvek x)
- pro každé $x, y \in M$ platí $\langle x, y \rangle \in A$ a $\langle y, x \rangle \in A$ (prvky x, y i y, x patří do stejné třídy rozkladu $\mathcal{M}$)
- pro každé $x, y, z \in M$ platí pokud $\langle x, y \rangle \in A$ a $\langle y, z \rangle \in A$ pak $\langle x, z \rangle \in A$ (jestliže y leží ve stejné třídě rozkladu $\mathcal{M}$ jako x a z, potom x leží ve stejné třídě rozkladu jako z).

Tedy každá relace ekvivalence je reflexivní, symetrická a tranzitivní.

Rozklad na množině M příslušný ekvivalenci A můžeme označovat M/A a nazývat faktorová množina M podle A. Faktorová množina M/A představuje zjednodušující pohled na výchozí množinu M, při kterém ztotožňujeme ty prvky M, které byly vzájemně zaměnitelné ekvivalencí A. Vezmeme-li množinu M a faktorovou množinu M/A, pak můžeme uvažovat zobrazení $f_A : M \to M/A$ a nazývat jej přirozené zobrazení. Srovnej s [2].

2. Popis programu

2.1. Knihovny a technologie použité při vývoji programu

2.1.1. WPF-Windows Presentation Foundation

Windows Presentation Foundation (WPF), dříve známá jako Avalon, je podmnožinou .NET Frameworku od verze 3.0, který používá značkovací jazyk XAML pro vytvoření uživatelsky bohatého rozhraní. Díky XAMLu jsou od sebe odděleny funkčnost a vzhled aplikace. Cílem WPF je sjednotit poutavé uživatelské rozhraní, 2D a 3D grafiku, vektorovou a rastrovou grafiku, animace, vázání dat a audio či video. WPF je přímým konkurentem pro starší WinForms. Všechna grafika (včetně samotných WPF oken) funguje pomocí Direct3D knihoven. Je zde možná spolupráce s WinForms pomocí tříd ElementHost a WindowsFormsHost. Třída ElementHost se využívá k hostování WPF elementu ve WinForms. Třída WindowsFormsHost se využívá k hostování WinForms prvku ve WPF. Viz [10].

2.1.2. Knihovna 3DTools

Tato knihovna vyvinuta WPF 3D týmem poskytuje užitečné metody pro práci v 3D prostoru ve WPF. Mezi tyto metody patří:

- použití interaktivních 2D prvků v 3D
- rotace kamery v 3D pomocí myši
- kreslení přímek v 3D.

2.2. Obecně o programu

Jedná se o interaktivní program na podporu výuky binárních relací. Program nabízí uživateli informace k tématu binárních relací a interaktivní příklady, které slouží k lepšímu pochopení těchto informací a na kterých si může uživatel ověřit své znalosti.

Program je napsán v jazyce C# a XAML ve vývojovém prostředí Visual Studio 2008. K tvorbě grafického uživatelského rozhraní programu byly použity technologie WinForms a WPF. Jazyk XAML (v kombinaci s jazykem C#) byl použit při tvorbě WPF komponent (např. komponenta sloužící k vykreslování 3D grafu, komponenta sloužící k vykreslení zobrazení v 3D prostoru). Tyto WPF komponenty jsou hostovány ve WinForms pomocí třídy ElementHost. Při návrhu a implementaci jsem dodržoval principy a metody objektově orientovaného programování. Dále jsem oddělil funkční část programu od jeho prezentace.

Program je otestovaný na operačním systému Windows XP. Ke své činnosti (minimální nároky programu) vyžaduje Windows XP Service Pack 2 a .NET

Framework 3.0. Instrukce pro instalaci a spuštění programu jsou uvedeny v přiloženém CD v souboru readme.txt.

2.3. Komponentová struktura programu

Program se skládá ze dvou částí. Knihovna LibBinaryRelation představuje jádro (funkční část programu). Aplikace BinaryRelation představuje rozhraní programu a využívá knihovnu LibBinaryRelation ke své činnosti.

2.3.1. Knihovna LibBinaryRelation

Představuje jádro programu. Tato knihovna využívá některé funkce knihovny 3DTools (kreslení 2D čar v 3D prostoru, interaktivita 2D prvků v 3D prostoru, rotace kamery v 3D prostoru užitím myši). Třídy, které tato knihovna poskytuje jsou:

- třídy pro reprezentaci množin a relací mezi množinami.

  `class Set;`
  Třída reprezentuje množinu. Důležité metody a vlastnosti této třídy jsou:

  - `Set(string name_set);`
    konstruktor třídy Set
  - `string NameSet;`
    vlastnost pro pojmenování nebo zjištění jména množiny
  - `bool AddItemToSet(string name_item);`
    `bool AddItemToSet(Brush item);`
    metody pro vložení prvku daného jména nebo dané barvy do množiny. Metody vrací informaci o tom, zda daný prvek byl úspěšně vložen do množiny nebo ne
  - `bool RemoveItem(string name_item);`
    `bool RemoveItem(Brush item);`
    metody pro vyjmutí prvku daného jména nebo barvy z množiny. Metody vrací informaci o tom, zda daný prvek byl úspěšně vyjmut z množiny nebo ne
  - `void RemoveAllItems();`
    metoda pro vyjmutí všech prvků množiny
  - `int CountItemsInSet();`
    vrací počet prvků v množině
  - `string GetNameItem(int number_item);`
    `Brush GetNameItem(int number_item);`
    parametr `number_item` udává pozici prvku v množině (kolikátý byl vložen do množiny)

`class Relation;`
Třída reprezentuje relaci mezi dvěma množinami. Metody a vlastnosti této třídy jsou:

- `Relation(name_relation);`
  konstruktor třídy Relation
- `string NameRelation;`
  vlastnost pro pojmenování, nebo zjištění jména relace
- `bool AddItemToRelation(int number_item1, int number_item2, Brush color_item);`
  `bool RemoveItemFromRelation(int number_item1, int number_item2);`
  metody pro vložení a vyjmutí prvku z dané relace. Metody vrací informaci o úspěšnosti operace. Parametry `number_item1, number_item2` představují pozice prvků v množinách, mezi kterými relaci uvažujeme
- `void RemoveAllItems();`
  metoda pro vyjmutí všech prvků z relace
- `int CountItemsInRelation();`
  vrací počet prvků v relaci
- `List<int> GetItemsInRelationWith(int number_item, int position_item);`
  parametr `position_item` specifikuje množinu a parametr `number_item` představuje pozici prvku v množině `position_item`. `position_item` může mít hodnotu 1 nebo 0. Metoda vrací pozice prvků, které jsou v relaci s prvkem `number_item`

- třídy pro výpočet související s relacemi (operace mezi relacemi, výpočet vlastností relací atd.).

  `class Computing`
  Statická třída pro výpočty související s relacemi. Metody této třídy jsou:

  - `Relation UnionRelations(string name_new_relation, params Relation[] relations);`
    Parametr `name_new_relation` určuje jméno relace vzniklé sjednocením relací daných parametrem `relations`
  - `Relation CompositionRelations(Relation r1, Relation r2, string name_new_relation);`
    vrací složení dvou relací `r1, r2`. Parametr `name_new_relation` určuje název nově vzniklé relace
  - `Relation InverseRelation(Relation r, string name_new_relation);`

vrací inverzní relaci k relaci `r`. Parametr `name_new_relation` určuje název nově vzniklé relace

- `Relation ReflexiveCRelation(Relation r, Set set, string name_new_relation);`
  `Relation SymmetryCRelation(Relation r, string name_new_relation);`
  `Relation TransitiveCRelation(Relation r, Set set, string name_new_relation);`
  vrací reflexivní, symetrický, tranzitivní uzávěr relace `r` na množině `set`. Parametr `name_new_relation` určuje název nově vzniklé relace
- `bool IsReflexive(Relation r, Set set);`
  `bool IsAntiSymmetry(Relation r);`
  `bool IsSymmetry(Relation r);`
  `bool IsTransitive(Relation r);`
  metody informují, zda relace `r` je reflexivní, antisymetrická, symetrická nebo tranzitivní
- `List<int> MinimalItem1(Relation r, Set set);`
  `List<int> MaximalItem1(Relation r, Set set);`
  vrací minimální (resp. maximální) prvky množiny `set`, která je uspořádaná relací `r`. Přesněji řečeno, metoda vrací seznam pozic prvků z množiny `set`
- `string SmallestItem(Relation r, Set set);`
  `string BiggestItem(Relation r, Set set);`
  vrací jméno nejmenšího (resp. největšího) prvku množiny `set`, která je uspořádaná relací `r`. Pokud takový prvek neexistuje, tak vrací null
- `Relation CreateRandomOrderRelation(Set set, string name_relation);`
  `Relation CreateRandomEquivalenceRelation(Set set, string name_relation);`
  vrací náhodnou relaci uspořádání (resp. ekvivalence) na množině `set`. Parametr `name_relation` určuje název nově vzniklé relace

- třídy pro výpočet a kreslení (množin, relací, grafů atd.)

  `class Drawing;`
  Statická třída pro kreslení. Důležité metody, které poskytuje jsou:

  - `void DrawArrow(Graphics g, PointF p1, PointF p2, float length_arrow,int width_line, Brush color_line, bool with_arrow,bool with_line);`
    vykresluje orientovanou hranu z bodu `p1` do bodu `p2`, `length_arrow` specifikuje délku čar, které tvoří šipku, `width_line` a `color_line`

určují pořadě tloušťku a barvu, kterou bude orientovaná hrana vykreslena, parametry `with_arrow` a `with_line` určují, zda bude vykreslena orientovaná hrana, neorientovaná hrana, nebo jen šipka

- `void DrawText(Graphics g, float x, float y, Font f, string text, char character, List<Brush> colors_items);`
  řetězec `text` bude vykreslen na kreslící povrch reprezentovaným parametrem `g`, každý výskyt znaku `character` v textovém řetězci `text` bude nahrazen kruhem o barvě specifikované v parametru `colors_items`. První výskyt znaku bude nahrazen kruhem s první barvou v seznamu barev, druhý výskyt znaku bude nahrazen kruhem s druhou barvou v seznamu atd. Dále objeví-li se v parametru `text` konstrukce např. `_prvek_jeho dolní index_`, pak bude slovo `jeho dolní index` vykresleno jako dolní index slova `prvek`. Nahradíme-li symboly `_` v předchozím příkladu za symbol `$` , pak bude slovo `jeho dolní index` vykresleno jako horní index slova `prvek`
- `void DrawSet(ScrollableControl control, Graphics g, RectangleF rect, Set set, Font f);`
  vykresluje množinu `set` na kreslící povrch reprezentovaným `g`, který náleží objektu `control`
- `void DrawRelation(ScrollableControl control, Graphics g1, Rectangle rect, Set set1, Set set2, Relation relation, Font font);`
  vykresluje relaci `relation` mezi množinami `set1`, `set2` na kreslící povrch reprezentovaným `g`, který náleží objektu `control`

- Třídy užitečné při práci v 3D prostoru ve WPF.

  `class Ball;`
  Třída reprezentuje kouli v 3D

  `class ScreenSpaceBezierCurve;`
  Třída reprezentuje bézierovu křivku v 3D

  `class Drawing3D;`
  Statická třída pro výpočet a kreslení v 3D. Některé metody této třídy jsou:

  - `List<Point3D> GetIntersectionBallAndLine(Point3D p_ball, double r_ball, Point3D a, Point3D b);`
    parametr `p_ball` určuje střed koule, `r_ball` určuje její poloměr, body `a`, `b` určují přímku. Metoda vrací seznam průsečíků dané přímky a koule.
  - `GeometryModel3D CreateTriangleModel(Point3D p0, Point3D p1, Point3D p2, Brush color_triangle);`
    metoda vrací trojúhelník barvy `color_triangle`, určený body `p0`, `p1`, `p2`.

- `Model3DGroup CreateArrowForLine(Point3D a, Point3D b, double length_arrow);`
  metoda vrací šipku pro přímku určenou body `a`, `b`. Parametr `length_arrow` určuje délku šipky
- `List<Ball> CreateBalls(List<Point3D> positions, List<Brush> brushes);`
  Metoda vrací seznam koulí o pozicích `positions` a barvách `brushes`

Dále knihovna poskytuje komponenty, které využívají předchozí třídy a jejich metody ke své realizaci.

- Komponenta `Graph` slouží k vykreslování 2D grafu. Důležité metody jsou:
  - `void SetGrid(int count_horizontal_lines, int count_vertical_lines);`
    nastavení mřížky. Pozice budoucích uzlů grafu jsou určeny průsečíky horizontálních a vertikálních čar. Počet těchto čar určují parametry `count_horizontal_lines`, `count_vertical_lines`. Tyto čáry jsou rozmístěny rovnoměrně na ploše celé komponenty.
  - `bool AddNode(string name, int number_h_line, int number_v_line);`
    `bool AddNode(Brush color, int number_h_line, int number_v_line);`
    metoda pro vložení uzlu s popiskem `name`, nebo o barvě `color` na pozici danou průsečíkem horizontální a vertikální čáry
  - `void AddArrow(int index_node1, int index_node2, Brush color_arrow);`
    vkládá orientovanou hranu o barvě `color_arrow` od uzlu `index_node1` k uzlu `index_node2`. Parametr `index_node1` (resp. `index_node2`) určuje uzel na základě pořadí, ve kterém byl vložen do grafu
  - `bool SetWidthAndColorArrow(int index_node1, int index_node2, int width, Brush color_arrow);`
    modifikuje tloušťku a barvu již vložené hrany
- Komponenta `Diagram` slouží k vykreslování 2D diagramu (Hasseova diagramu). Důležité metody jsou:
  - `void SetGrid(int count_horizontal_lines, int count_vertical_lines);`
    nastavení mřížky. Pozice budoucích uzlů grafu jsou určeny průsečíky horizontálních a vertikálních čar. Počet těchto čar určují parametry `count_horizontal_lines`, `count_vertical_lines`. Tyto čáry jsou rozmístěny rovnoměrně na ploše celé komponenty

  - `void AddNode(string name_node, float x_position, float y_position, bool node_is_visible);`
    metoda vkládá uzel do diagramu
  - `bool AddArrow(string name_node1, string name_node2);`
    metoda vkládá neorientovanou hranu mezi uzly o jménech `name_node1`, `name_node2`
  - `void SetAsHasseDiagram(Set set, Relation order_relation, int w);`
    vykreslí Hasseův diagram uspořádané množiny `set` podle relace `order_relation`. Parametr `w` určuje max. počet uzlů na řádku

- Komponenta `Matrix` slouží k zobrazení maticové reprezentace binární relace. Důležité metody jsou:
  - `void AddNamesItems(List<string> rows, List<string> columns, string name_matrix);`
    specifikuje počet řádků a sloupců matice. Jednotlivé řádky a sloupce budou popsány položkami parametrů `rows` a `columns`. Parametr `name_matrix` určuje jméno matice
  - `void PutValueToItem(int row, int column, string value, Color color_item);`
    vloží hodnotu `value` o barvě `color_item` na pozici, která je průsečíkem řádku o indexu `row` a sloupce o indexu `column` matice

- WPF komponenta `Graph1` slouží k vykreslování 3D grafu. Důležité metody jsou:
  - `void AddNodes(List<Brush> colors_nodes);`
    vkládá uzly o barvách `color_nodes`
  - `void AddArrow(int first_node, int second_node, Color color);`
    `void RemoveArrow(int first_node, int second_node);`
    metody pro vložení a vyjmutí orientované hrany mezi uzly `first_node`, `second_node`. Parametr `first_node` (resp. `second_node`) určuje uzel na základě pořadí, ve kterém byl vložen do grafu

Další komponenty knihovny popisují a znázorňují některé důležité pojmy související s binárními relacemi. Tyto komponenty slouží k interaktivní výuce binárních relací. Mezi ně patří:

`BinaryRelationOnSet`	`CartesianProduct`
`RepreRelationGraphX`	`RepreRelationMatrix`
`RepreRelationSetGraphMatrix`	`UnionRelation`
`IntersectionRelation`	`InverseRelation`
`TypeOfFunction`	`FunctionAndRelationX`
`ReflexiveRelation`	`SymmetryRelation`
`TransitiveRelation`	`Closures`
`EquivalenceRelation`	`OrderRelation1`
`OrderRelation2`	`OrderRelation3`

2.3.2. Aplikace BinaryRelation

Aplikace využívá ke své činnosti třídy a komponenty knihovny LibBinaryRelation. Uživatelské rozhraní aplikace je tvořeno *hlavním oknem* a oknem zobrazujícím obsah (*obsahové okno*), viz obrázek 11.

Obsahové okno je rozděleno do několika sekcí (např. sekce Reprezentace relace, sekce Operace s relacemi atd.). Každá sekce obsahuje položku *Text* a podsekci *Příklady*. Po stisknutí položky *Text* se zobrazí v *hlavním okně* text, který vysvětluje pojmy související s danou sekcí. Po stisknutí položky obsažené v podsekci *Příklady* se v *hlavním okně* zobrazí interaktivní příklad související s danou sekcí a stisknutou položkou.

V pravém horním rohu ukotveného *obsahového okna* jsou umístěny položky sloužící k manipulaci s oknem. Mezi ně patří:

- *Zavřít*
  slouží k zavření okna. Ke znovuotevření poté musíme použít položku *Obsah* v menu *hlavního okna*

- *Skrýt*
  slouží k nastavení okna na automatické skrytí, nebo naopak pevné zobrazení

- *Volby-Plovoucí*
  pomocí ní lze zrušit ukotvení okna. Poté lze okno znovu ukotvit k libovolné vnitřní straně *hlavního okna*. Při stisknutí levého tlačítka myši nad horní lištou neukotveného okna a tažením libovolným směrem se zobrazí možnosti ukotvení

Hlavní okno je rozděleno do tří částí:

- *Hlavní menu*
  položky tohoto menu jsou:

  - *Obsah*
    zobrazí *obsahové okno* aplikace

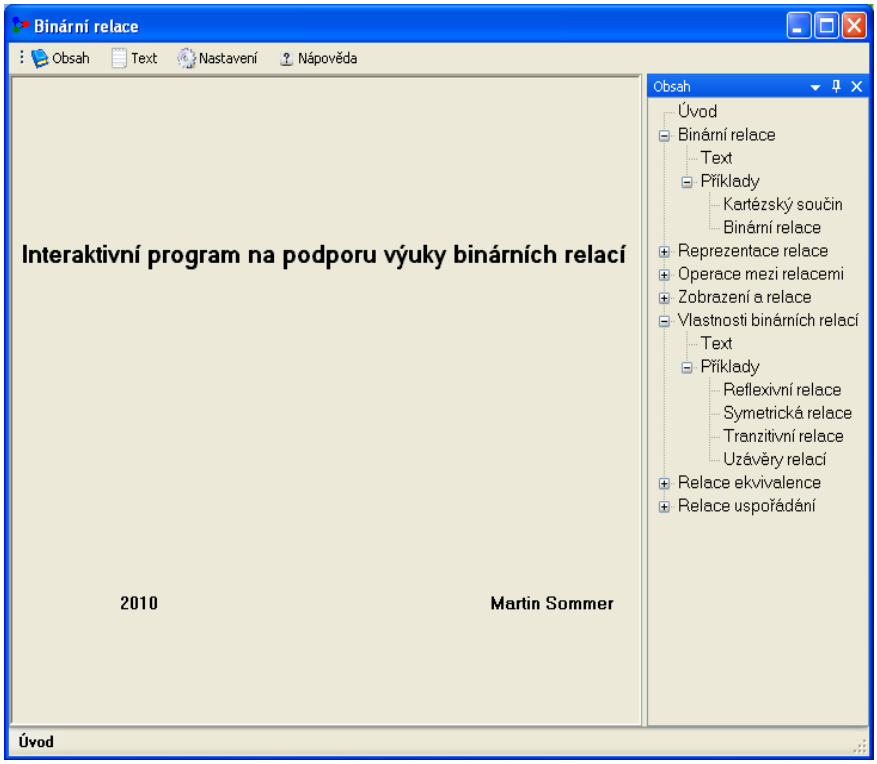
– *Text*
pomocí ní lze přistoupit ke stejným textům, které jsou popsány v odstavci o *obsahovém okně*

– *Nastavení*
zpřístupňuje položky pomocí nichž lze nastavit font textu a barvu pozadí příkladů a *obsahového okna*

– *Nápověda*
poskytuje základní informace a možnosti aplikace

- Okno pro zobrazení textů a příkladů, které jsme vybrali v *obsahovém okně*

- Lišta, která informuje uživatele o aktuálně zobrazeném textu nebo příkladu



Obrázek 11. Aplikace BinaryRelation

Příklady slouží k interaktivní výuce pojmů souvisejících s binárními relacemi. Jednotlivé příklady obsahují prvky, které zajišťují interakci uživatele s příkladem. Mezi ně patří:

- menu s položkami *Úprava množiny* a *Úprava relace*, viz obrázek 12
menu se nachází v horní části příkladu. Při stisknutí levého tlačítka myši

Obrázek 12. Menu s položkami *Úprava množiny* a *Úprava relace*

nad položkou *Úprava množiny* (resp. *Úprava relace*) se zobrazí formulář pro úpravu množiny (resp. relace)

- prvek pro výběr hodnoty z množiny, viz obrázek 13
  prvek obsahuje tlačítka sloužící k výběru hodnoty z množiny a pole, které zobrazuje vybranou hodnotu

Obrázek 13. Prvek pro výběr hodnoty z množiny

- prvek pro výběr uspořádané dvojice, viz obrázek 14
  prvek obsahuje tlačítka pro výběr první a druhé hodnoty uspořádané dvojice a dvě pole, která zobrazují vybrané hodnoty

Obrázek 14. Prvek pro výběr uspořádané dvojice

- tlačítka pro manipulaci s hodnotou vybranou prvkem pro výběr hodnoty z množiny a tlačítka pro manipulaci s uspořádanou dvojicí vybranou prvkem pro výběr uspořádané dvojice
- formulář pro úpravu množiny
  obsahuje prvek pro výběr hodnoty z množiny a tlačítka, pomocí nichž lze vložit do množiny nebo vyjmout z množiny vybranou hodnotu
- formulář pro úpravu relace
  obsahuje prvek pro výběr uspořádané dvojice a tlačítka, pomocí nichž lze vložit do relace nebo vyjmout z relace vybranou uspořádanou dvojici

V příkladu *Reprezentace relace pomocí grafu* (resp. *Kartézský součin*) lze rotovat prvky umístěné v položce *Graf reprezentující relaci* (resp. *Prvky množiny*). Rotace se provede při stisknutí levého tlačítka myši nad jednou z položek a táhnutím myši libovolným směrem.

Závěr

Prostudoval jsem binární relace. V teoretické části práce jsou popsány nejdůležitější pojmy, které souvisí s binárními relacemi. Poté jsem vytvořil interaktivní program na podporu výuky binárních relací. Při návrhu a implementaci jsem dodržoval principy a metody objektově orientovaného programování. Program se skládá z knihovny LibBinaryRelation a aplikace BinaryRelation. V praktické části této práce jsou popsány nejdůležitější třídy a komponenty knihovny LibBinaryRelation a vysvětleno ovládání aplikace BinaryRelation. Program je napsán v jazyce C# a XAML ve vývojovém prostředí Visual Studio 2008.

Doufám, že mnou vytvořená práce bude užitečná komukoli, kdo by se rád seznámil se základními pojmy tématu binární relace.

Conclusions

I have studied the topic of binary relation. In the theoritical part of my work I describe the most important terms connected with the binary relation. I continued to create an interactive programme to support the teaching and learning of binary relation. When designing and implementing the programme I tried to follow the principles and methods of the object-oriented programming. The programme consists of the library LibBinaryRelation and the application BinaryRelation. In the practical part of this work I describe the most important classes and components of the library LibBinaryRelation and I explain the operating of the application BinaryRelation. The programme is written in the languages C# and XAML in the development enviroment Visual Studio 2008.

I hope that my work will be useful for anyone who would be interested to get to know the basic concepts regarding the topic of binary relation.

Reference

[1] Bělohlávek R., Vychodil V.: *Diskrétní matematika pro informatiky 1.* Katedra informatiky, Univerzita Palackého, Přírodovědecká fakulta, 2006, http://phoenix.inf.upol.cz/ kolarik/XDIMA/DiskretniMatematika1.pdf

[2] Bělohlávek R., Vychodil V.: *Diskrétní matematika pro informatiky 2.* Katedra informatiky, Univerzita Palackého, Přírodovědecká fakulta, 2006, http://phoenix.inf.upol.cz/ kolarik/XDIMA/DiskretniMatematika2.pdf

[3] Hlíněný P.: *Uspořádané množiny a uzávěry.* Slajdy k tématu uspořádané množiny a uzávěry

[4] Petzold Ch.: *Programování Microsoft Windows v jazyce C#, svazek 1.* SoftPress, Praha, 2003

[5] Petzold Ch.: *Programování Microsoft Windows v jazyce C#, svazek 2.* SoftPress, Praha, 2003

[6] Petzold Ch.: *Mistrovství ve Windows Presentation Foundation.* Computer press, 2008

[7] Šrejder J. A.: *Binární relace.* SNTL, Praha, 1978

[8] Šturala A.: *Začínáme s WPF.* Seriál článků o technologii WPF, 2007, http://www.vyvojar.cz/Series/3-zaciname-s-wpf.aspx

[9] Virius M.: *Od C++ k C#.* Kopp, České Budějovice, 2002

[10] *Wikipedie, otevřená encyklopedie,* http://cs.wikipedia.org

C. Obsah přiloženého CD

Na přiloženém CD jsou uloženy kompletní zdrojové texty programu, text této dokumentace, instalátor programu a přeložený program, který lze spustit přímo z CD.

Následuje adresářová struktura CD a stručný popis obsahu každého adresáře:

`bin/`
: Podadresář `installer/` obsahuje instalátor programu. Podadresář `program/` obsahuje soubory přeloženého programu, spustitelné přímo z CD a všechny potřebné knihovny.

`doc/`
: Dokumentace této práce ve formátech PostScript, PDF a soubory nutné pro vygenerování PDF souboru.

`src/`
: Zdrojové texty programu. Podadresář `BinaryRelation/` obsahuje zdrojové texty aplikace BinaryRelation. Podadresář `LibBinaryRelation/` obsahuje zdrojové texty knihovny LibBinaryRelation. Podadresář `installer/` obsahuje zdrojové texty instalátoru.

`readme.txt`
: Instrukce pro instalaci a spuštění programu. Požadavky na provoz programu.