PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

DIPLOMOVÁ PRÁCE

Formální konceptuální analýza nad neúplnými daty

2014 Bc. Martin Kauer

Anotace

Problémem neúplných dat se zabývá stále více článků z různých oblastí informatiky a to včetně formální konceptuální analýzy (FKA). Tento problém nabývá na důležitosti, protože je nutné zpracovávat větší objemy dat a ne vždy jsou úplná data k dispozici. Práce obsahuje úvod do problematiky neúplných dat z pohledu FKA a shrnutí několika metod pro práci s takovými daty. Dále představuje několik vlastních teoretických výsledků z této oblasti, na základě kterých jsem vytvořil algoritmy pro zúplnění neúplných dat a transformaci konceptuálních svazů příslušných možným zúplněním.

Děkuji doc. RNDr. Michalu Krupkovi, Ph.D. za vedení této diplomové práce, přínosné konzultace, pomoc s formalizací představených problémů a hlavně za trpělivost, kterou se mnou měl. Dále děkuji Mgr. Janu Laštovičkovi za poskytnuté rady a materiály.

Obsah

Seznam tabulek

Úvod

V reálných aplikacích se neúplná data vyskytují běžně a je nutné s nimi pracovat. Základní metody zpracování dat jsou ve většině případů navrženy pro data úplná a neúplná data je třeba předzpracovat nebo zpracovat zcela jinak. Předzpracování většinou spočívá v odstranění neúplných záznamů nebo v jejich zúplnění. Na zúplnění dat existuje několik metod, které odhadují neznámé hodnoty na základě známých dat nebo s pomocí znalostí uživatele. Výjimkou není ani formální konceptuální analýza (FKA), jejíž základní metody pracují výhradně s úplnými daty. Cílem této práce je popsat vybrané přístupy k neúplným datům ve FKA a přinést nové výsledky z této oblasti.

První kapitola zpracovává stručný úvod do FKA. Obsahuje potřebné definice a pojmy k pochopení dalších kapitol. Druhá kapitola je věnována shrnutí několika vybraných článků věnujících se problematice neúplných dat ve FKA. Zejména se jedná o zúplňování dat pomocí atributových implikací, explorativní analýzy a maticové faktorizace.

Poslední kapitola obsahuje vlastní výsledky. Prvním z těchto výsledků je analýza experimentální metody zúplňování dat pomocí počítání konceptů. Pro tuto metodu bylo nutné nejdříve vytvořit algoritmus, pomocí kterého jsem provedl sadu experimentů. Druhým výsledkem je analýza změny konceptuálního svazu při odebrání právě jedné incidence. Konečným výsledkem je algoritmus pro transformaci konceptuálního svazu při odebrání incidence.

1. Formální konceptuální analýza

Formální konceptuální analýza je metoda analýzy dat, která byla představena ve článku Willeho [1]. Předmětem zkoumání FKA je analýza dat popisujících vztah mezi danou množinou objektů a danou množinou atributů. Jedním ze základních výstupů FKA je konceptuální svaz, což je množina formálních konceptů spolu s jejich uspořádáním. Formální koncept je shluk v datech daný množinou objektů a množinou atributů. Formální koncepty můžeme mezi sebou porovnávat a pokud jsou srovnatelné, tak řekneme, že jeden je obecnější než druhý nebo naopak konkrétnější. Důležitou vlastností konceptuálního svazu je jeho snadná grafická reprezentace. Uživatel má možnost data zkoumat v grafické podobě, která v sobě zahrnuje i hierarchické uspořádání. Dalším ze základních výstupů FKA je množina platných atributových implikací ve vstupních datech. Atributová implikace popisuje určitou závislost mezi atributy vstupních dat.

V této kapitole jsou uvedeny základní pojmy a výsledky z FKA, které jsou potřebné k pochopení dalších částí této práce. Při jejím zpracování jsem vycházel z elektronického učebního textu Bělohlávka [2] a z knihy Gantera a Willeho [3]. Většina uvedených vět je ponechána bez důkazů, protože to není předmětem této práce. Zájemcům o hlubší studium FKA doporučuji zmíněné zdroje.

1.1. Základy formální konceptuální analýzy

Základním pojmem ve FKA je *formální kontext*, který představuje vstupní data. Metody FKA pracují s jednoduchými binárními daty, která se skládají z množiny objektů, množiny atributů a vztahem určujícím, zda daný objekt má daný atribut.

Definice 1.1 (formální kontext)**.**
Formální kontext je trojice $\langle X, Y, I \rangle$, kde X a Y jsou neprázdné množiny a I je binární relace mezi těmito množinami ($I \subseteq X \times Y$).

V předchozí definici interpretujeme X jako množinu objektů, Y jako množinu atributů a $\langle x, y \rangle \in I$ má význam „objekt x má atribut y“.

Formální kontext lze reprezentovat pomocí tabulky, kde záhlaví řádků jsou jednotlivé objekty a záhlaví sloupců jednotlivé atributy. V daném poli tabulky je buď křížek (objekt má příslušný atribut) nebo prázdné pole (objekt nemá příslušný atribut). Na pořadí řádků a sloupců nezáleží. Pokud v tabulce příslušné danému formálnímu kontextu prohodíme některé řádky/sloupce, pořád bude reprezentovat tentýž formální kontext.

Není náhoda, že každý formální kontext lze reprezentovat tabulkou. Tabulka představuje přehledný zápis jednoduchých binárních dat, která chceme pomocí FKA analyzovat. Můžeme tedy používat pojem *tabulka* (příslušná formálnímu kontextu) a víme, že se jedná o reprezentaci určitého formálního kontextu. Tabulka je také vhodnější pro grafické znázornění i pro lidskou představivost.

I	y_1	y_2	y_3	y_4	y_5
x_1		×		×	×
x_2	×		×	×	×
x_3	×			×	
x_4	×	×			×
x_5		×	×	×	

Formální kontext 1: Reprezentace formálního kontextu tabulkou.

V následujícím textu budeme vždy uvažovat formální kontext $C = \langle X, Y, I \rangle$ a k němu příslušný konceptuální svaz $\langle \mathcal{B}(X, Y, I), \leq \rangle$, pokud nebude dáno jinak. Dále zavedeme základní operátory FKA.

Definice 1.2 (šipkové operátory)**.**
Pro množinu $A \subseteq X$ a $B \subseteq Y$ definujeme

$$\begin{aligned} A^{\uparrow_I} &= \{y \in Y \mid \textit{pro všechna } x \in A \textit{ platí } \langle x, y \rangle \in I\}, \\ B^{\downarrow_I} &= \{x \in X \mid \textit{pro všechna } y \in B \textit{ platí } \langle x, y \rangle \in I\}. \end{aligned}$$

Operátory $^{\uparrow_I}, ^{\downarrow_I}$ jsou tzv. *šipkové operátory*. Operátor $^{\uparrow_I}$ přiřazuje množině vstupních objektů množinu všech atributů, které tyto objekty sdílí. Podobně operátor $^{\downarrow_I}$ přiřazuje množině vstupních atributů množinu všech objektů, které mají tyto atributy. Pokud je jasně daný formální kontext, můžeme místo $^{\uparrow_I}, ^{\downarrow_I}$ psát pouze $^{\uparrow}, ^{\downarrow}$.

Matematické základy formální konceptuální analýzy se opírají zejména o *Galoisovy konexe*, *uzávěrové operátory* a příslušné *uzávěrové systémy*. Je vhodné tyto pojmy dobře znát, společně s jejich základními vlastnostmi. V tomto textu je uvedena pouze malá část těchto vlastností, která je potřeba k pochopení dalších částí práce.

Věta 1.3.
Nechť $C = \langle X, Y, I \rangle$ je formální kontext, pak operátory $\langle ^{\uparrow_I}, ^{\downarrow_I} \rangle$, indukované kontextem C, tvoří antitonní Galoisovy konexe, tzn. že platí:

1. $A_1 \subseteq A_2 \implies {A_2}^{\uparrow_I} \subseteq {A_1}^{\uparrow_I}$ *- antitonie* $^{\uparrow_I}$,
2. $B_1 \subseteq B_2 \implies {B_2}^{\downarrow_I} \subseteq {B_1}^{\downarrow_I}$ *- antitonie* $^{\downarrow_I}$,
3. $A \subseteq A^{\uparrow_I\downarrow_I}$ *- extenzivita* $^{\uparrow_I\downarrow_I}$,
4. $B \subseteq B^{\downarrow_I\uparrow_I}$ *- extenzivita* $^{\downarrow_I\uparrow_I}$.

Galoisovy konexe jsou důležitým matematickým pojmem, který je znám již dlouho. Jelikož šipkové operátory tvoří, dle předchozí věty, Galoisovy konexe, můžeme využívat všech jejich známých vlastností. Některé z těchto vlastností

jsou představeny v následujícím textu. Přestože se jedná o vlastnosti Galoisových konexí obecně, jsou představeny konkrétně na výše zavedených šipkových operátorech, a to kvůli přehlednosti. Prvním důležitým pojmem je *množina pevných bodů* Galoisových konexí.

Definice 1.4 (pevné body Galoisových konexí)**.**
Pro Galoisovy konexe $\langle {}^{\uparrow}, {}^{\downarrow} \rangle$ *definujeme množinu pevných bodů jako*

$$fix(\langle {}^{\uparrow}, {}^{\downarrow} \rangle) = \{\langle A, B \rangle \in 2^X \times 2^Y \mid A^{\uparrow} = B,\ B^{\downarrow} = A\}.$$

Další známou vlastností Galoisových konexí je tzv. *řetězení*. Tato vlastnost nám umožňuje určitým způsobem „krátit“ několikanásobnou aplikaci Galoisových konexí. Řetězení můžeme formalizovat následovně.

Věta 1.5.
Pro Galoisovy konexe $\langle {}^{\uparrow}, {}^{\downarrow} \rangle$ *a libovolné* $A \subseteq X,\ B \subseteq Y$ *platí*

$$A^{\uparrow} = A^{\uparrow\downarrow\uparrow},\ B^{\downarrow} = B^{\downarrow\uparrow\downarrow}.$$

Pojem *formální koncept* je jedním ze základních pojmů FKA. Na otázku, co koncept je, lze odpovědět několika způsoby. V našem případě vycházíme z pojetí *Port–Royalské logiky*, kde je koncept definován *extentem* a *intentem*. *Extent* je množina objektů zahrnutých daným konceptem. *Intent* je množina atributů zahrnutých daným konceptem.

Definice 1.6 (formální koncept)**.**
Formální koncept nad $\langle X, Y, I \rangle$ *je dvojice* $\langle A, B \rangle$*, kde* $A \subseteq X,\ B \subseteq Y$ *takové, že*

$$A^{\uparrow} = B \quad a \quad B^{\downarrow} = A.$$

Množina objektů A *se nazývá extent, množina atributů* B *se nazývá intent.*

Slovně vyjádřeno, $\langle A, B \rangle$ je formální koncept, právě když A je množina všech objektů sdílejících všechny atributy z B a B je množina všech atributů, které mají všechny objekty z A. Je dobré si uvědomit, že ne každá podmnožina objektů je extentem (obdobně pro atributy). Důležitá je také geometrická interpretace formálního konceptu. Z tohoto pohledu jsou koncepty právě maximální obdélníky v tabulce reprezentující daný formální kontext, přičemž je možné řádky a sloupce libovolně prohazovat. Geometrická interpretace je vhodná pro lidskou představivost a mnohdy je lepší uvažovat nad koncepty jako nad maximálními obdélníky. Z pohledu operátorů ${}^{\uparrow}, {}^{\downarrow}$ je $\langle A, B \rangle$ formální koncept, právě když $\langle A, B \rangle$ je pevný bod $\langle {}^{\uparrow}, {}^{\downarrow} \rangle$.

Nyní máme zavedené formální koncepty, ale nemáme určené jejich uspořádání, které potřebujeme k definici konceptuálního svazu. Přirozeným požadavkem na takové uspořádání je možnost jej interpretovat jako uspořádání dle obecnosti konceptů. Při splnění tohoto požadavku potom $\langle A_1, B_1 \rangle \leq \langle A_2, B_2 \rangle$ znamená, že $\langle A_1, B_1 \rangle$ je specifičtější než $\langle A_2, B_2 \rangle$ (ekvivalentně, $\langle A_2, B_2 \rangle$ je obecnější než $\langle A_1, B_1 \rangle$).

Definice 1.7 (uspořádání formálních konceptů)**.**
Pro formální koncepty $\langle A_1, B_1\rangle$, $\langle A_2, B_2\rangle$ *definujeme*

$$\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle, \text{ právě když } A_1 \subseteq A_2 \text{ (ekvivalentně, } B_2 \subseteq B_1\text{).}$$

Množina všech formálních konceptů, nad daným formálním kontextem, společně s výše definovaným uspořádáním tvoří tzv. *konceptuální svaz.*

Definice 1.8 (konceptuální svaz)**.**
Označme $\mathcal{B}(X, Y, I)$ *množinu všech formálních konceptů nad* $\langle X, Y, I\rangle$*, potom* $\langle \mathcal{B}(X, Y, I), \leq\rangle$ *se nazývá konceptuální svaz příslušný formálnímu kontextu* $\langle X, Y, I\rangle$*.*

Konceptuální svaz $\langle \mathcal{B}(X, Y, I), \leq\rangle$ reprezentuje množinu všech shluků dat (formálních konceptů) ve formálním kontextu $\langle X, Y, I\rangle$. Konceptuální svaz lze dobře graficky reprezentovat a poskytuje uživateli grafický vhled do vstupních dat reprezentovaných daným formálním kontextem.

Věta 1.9 (složená zobrazení $^{\uparrow\downarrow}$, $^{\downarrow\uparrow}$)**.**
Složené zobrazení $^{\uparrow\downarrow}$ *je uzávěrový operátor na* X *(obdobně pro složené zobrazení* $^{\downarrow\uparrow}$*), tzn. že platí:*

1. $A \subseteq A^{\uparrow\downarrow}$ *- přímo z definice Galoisových konexí,*
2. $A_1 \subseteq A_2 \implies {A_2}^{\uparrow} \subseteq {A_1}^{\uparrow} \implies A_1^{\uparrow\downarrow} \subseteq {A_2}^{\uparrow}$ *- z antitonie* $^{\uparrow}, ^{\downarrow}$*,*
3. $A^{\uparrow\downarrow} = A^{\uparrow\downarrow\uparrow\downarrow}$ *- z vlastnosti řetězení.*

Předchozí výsledek opět vychází pouze z vlastností Galoisových konexí a nejedná se o zvláštní vlastnost šipkových operátorů.

Před uvedením tzv. *základní věty o konceptuálních svazech* připomeňme pojem *supremálně (infimálně) husté množiny.* Na prvky těchto množin se můžeme dívat, jako na stavební bloky úplného svazu $\mathbf{V} = \langle V, \leq\rangle$. Všechny prvky V lze získat jako suprema (infima) prvků supremálně (infimálně) husté množiny.

Poznámka 1.10 (supremálně, infimálně husté množiny)**.**
Nechť $\mathbf{V} = \langle V, \leq\rangle$ *je úplný svaz, pak množina* $K \subseteq V$ *je:*

1. $\bigvee$*-hustá, právě když* $\forall v \in V,\ \exists K' \subseteq K :\ v = \bigvee K$*,*
2. $\bigwedge$*-hustá, právě když* $\forall v \in V,\ \exists K' \subseteq K :\ v = \bigwedge K$*.*

Věta 1.11 (základní věta o konceptuálních svazech)**.**
Větu lze rozdělit do dvou následujících částí:

1. $\langle \mathcal{B}(X,Y,I), \leq \rangle$ *je úplný svaz, kde suprema a infima jsou dána*

$$\bigwedge_{j\in J} \langle A_j, B_j \rangle = \left\langle \bigcap_{j\in J} A_j, \left(\bigcup_{j\in J} B_j \right)^{\downarrow_I \uparrow_I} \right\rangle,$$
$$\bigvee_{j\in J} \langle A_j, B_j \rangle = \left\langle \left(\bigcup_{j\in J} A_j \right)^{\uparrow_I \downarrow_I}, \bigcap_{j\in J} B_j \right\rangle.$$

2. *Libovolný úplný svaz* $\mathbf{V} = \langle V, \leq \rangle$ *je izomorfní s* $\langle \mathcal{B}(X,Y,I), \leq \rangle$*, právě když existují zobrazení* $\gamma_I : X \mapsto V$*,* $\mu_I : Y \mapsto V$ *taková že platí:*

   (a) $\gamma_I(X)$ *je* $\bigvee$*-hustá,* $\mu_I(X)$ *je* $\bigwedge$*-hustá,*

   (b) $\gamma_I(x) \leq \mu_I(y)$*, právě když* $\langle x, y \rangle \in I$*.*

Jedním z užitečných důsledků faktu, že složením šipkových operátorů získáme uzávěrový operátor, je schopnost jednoduše vyjádřit nejmenší extent (intent) obsahující danou množinu objektů (atributů).

Věta 1.12.
Nechť $A \subseteq X$*. Platí* $A^{\uparrow\downarrow}$ *je nejmenší extent obsahující* A*.*
Nechť $B \subseteq Y$*. Platí* $B^{\downarrow\uparrow}$ *je nejmenší intent obsahující* B*.*

Důkaz. Jestliže C je extent takový, že $A \subseteq C$, pak z monotonie uzávěrového operátoru $^{\uparrow\downarrow}$ platí $A^{\uparrow\downarrow} \subseteq C^{\uparrow\downarrow}$.

Obdobně, jestliže D je intent takový, že $B \subseteq D$, pak z monotonie uzávěrového operátoru $^{\downarrow\uparrow}$ platí $B^{\downarrow\uparrow} \subseteq D^{\downarrow\uparrow}$. □

1.2. Klarifikace a redukce formálního kontextu

Definice formálního kontextu (1.1) dovoluje například více objektů se stejnou množinou atributů. V tabulce si tento případ lze představit jako dva stejné řádky (mimo záhlaví). Toto pozorování lze provést i pro atributy. Otázkou je, jak tato situace ovlivní příslušný konceptuální svaz.

Definice 1.13 (klarifikovaný formální kontext)**.**
Formální kontext $\langle X, Y, I \rangle$ *nazveme klarifikovaný, pokud tabulka příslušná tomuto kontextu neobsahuje identické řádky nebo sloupce.*

Klarifikací formálního kontextu nazveme proces odstranění duplicitních řádků a sloupců z příslušné tabulky. Klarifikací získáme novou tabulku, která odpovídá klarifikovanému formálnímu kontextu.

Věta 1.14.
Nechť formální kontext $\langle X_k, Y_k, I_k \rangle$ vznikne klarifikací formálního kontextu $\langle X, Y, I \rangle$. Potom konceptuální svaz $\langle \mathcal{B}(X_k, Y_k, I_k), \leq \rangle$ je izomorfní s konceptuálním svazem $\langle \mathcal{B}(X, Y, I), \leq \rangle$.

Klarifikace formálního kontextu nemá vliv (až na značení) na příslušný konceptuální svaz. To ale neznamená, že nemá vliv na samotné koncepty. Odstranění duplicitních řádků a sloupců vlastně znamená odstranění příslušných objektů a atributů, což koncepty ovlivní. Další úpravou formálního kontextu, která nemá vliv (až na značení) na konceptuální svaz, je *redukce*.

Definice 1.15 (reducibilní atributy/objekty)**.**
Atribut $y \in Y$ nazveme reducibilní, právě když

$$\exists Y' \subset Y,\ y \notin Y' :\ \{y\}^{\downarrow} = \bigcap_{z \in Y'} \{z\}^{\downarrow}.$$

Objekt $x \in X$ nazveme reducibilní, právě když

$$\exists X' \subset X,\ x \notin X' :\ \{x\}^{\uparrow} = \bigcap_{z \in X'} \{z\}^{\uparrow}.$$

Reducibilní atributy jsou právě ty, jejichž sloupce v tabulce lze získat průnikem jiných sloupců. Podobně *reducibilní objekty* jsou právě ty, jejichž řádky v tabulce lze získat průnikem jiných řádků. Reducibilita je úzce spojena se supremální (infimální) reducibilitou v $\langle \mathcal{B}(X, Y, I), \leq \rangle$. Pomocí těchto pojmů lze definovat redukovaný formální kontext.

Definice 1.16 (redukovaný formální kontext)**.**
Formální kontext nazveme redukovaný, pokud neobsahuje reducibilní objekty a atributy.

Redukcí formálního kontextu nazveme proces odstranění reducibilních objektů a atributů. Podobně jako u klarifikace, redukce kontextu nemá vliv na jeho konceptuální svaz (až na značení), ale na samotné koncepty ano.

Věta 1.17.
Nechť atribut $y \in Y$ je reducibilní. Potom $\langle \mathcal{B}(X, Y \setminus \{y\}, J), \leq \rangle$ je izomorfní s $\langle \mathcal{B}(X, Y, I), \leq \rangle$, kde J je restrikce I na $Y \setminus \{y\}$.
Nechť objekt $x \in X$ je reducibilní. Potom $\langle \mathcal{B}(X \setminus \{x\}, Y, J), \leq \rangle$ je izomorfní s $\langle \mathcal{B}(X, Y, I), \leq \rangle$, kde J je restrikce I na $X \setminus \{x\}$.

1.3. Vícehodnotové formální kontexty

Rozšířením základní definice formálního kontextu je možné definovat tzv. *vícehodnotové formální kontexty*. Ty si můžeme představit jako tabulky, které v jednotlivých polích mohou mít i jiné hodnoty než křížek.

Definice 1.18 (vícehodnotový formální kontext)**.**
Vícehodnotový formální kontext je čtveřice $\langle X, Y, V, I\rangle$*, kde* X *je neprázdná množina objektů,* Y *je neprázdná množina atributů,* V *je množina všech přípustných hodnot atributů z* Y *a* I *je relace mezi těmito třemi množinami (*$I \subseteq X \times Y \times V$*) taková, že platí*

$$(\langle x, y, v\rangle \in I) \wedge (\langle x, y, w\rangle \in I) \implies v = w.$$

Vícehodnotový formální kontext umožňuje práci s vícehodnotovými atributy. Neuvažujeme pouze zda má objekt daný atribut, ale uvažujeme i nad příslušnou hodnotou atributu. Přirozeným omezením na vícehodnotový formální kontext je požadavek, aby každé dvojici objektu a atributu byla přiřazena nejvýše jedna přípustná hodnota.

Analogicky, jako v případě binárních formálních kontextů, lze vícehodnotový formální kontext $\langle X, Y, V, I\rangle$ reprezentovat tabulkou, kde záhlaví řádků jsou jednotlivé objekty z X a záhlaví sloupců jednotlivé atributy z Y. Rozdíl je u hodnot v jednotlivých polích tabulky. V tomto případě je v daném poli tabulky určeném objektem x a atributem y hodnota $v \in V$, pokud $\langle x, y, v\rangle \in V$, jinak prázdno.

Pro vícehodnotové formální kontexty je vhodné zavést pojem *doména atributu*. Tento pojem je analogií aktivní domény z relačního databázového modelu.

Definice 1.19 (doména a škála atributu)**.**
Nechť $\langle X, Y, V, I\rangle$ *je vícehodnotový formální kontext. Doménu atributu* $y \in Y$ *definujeme jako*

$$D_y = \{v \mid \exists x \in X : \langle x, y, v\rangle \in I\}.$$

Škála pro atribut $y \in Y$ *je binární formální kontext* $S_y = \langle X_y, Y_y, I_y\rangle$ *takový, že* $D_y \subseteq X_y$*.*

Máme definovaný pojem vícehodnotového formálního kontextu, ale otázkou je, jak takový formální kontext zpracovat. V předchozích definicích jsme vždy pracovali pouze s binárním formálním kontextem. Jednou z možností, jak tento problém vyřešit, je vhodným způsobem převést vícehodnotový formální kontext na binární. Toho lze dosáhnout několika způsoby. Uvedeme si zde jeden ze základních způsobů, tzv. *jednoduché škálování*.

	Výkon (HP)	Barva	Cena ($)
Porshe 911 GT3	475	černá	130 400
Nissan GT-R	545	bílá	115 710
Pagani Huayra	720	šedá	1 424 966

Formální kontext 2: Reprezentace vícehodnotového formálního kontextu tabulkou.

Definice 1.20 (jednoduché škálování)**.**
Nechť $\langle X, Y, V, I\rangle$ je vícehodnotový formální kontext a $S_y = \langle X_y, Y_y, I_y\rangle$ škály pro všechny atributy $y \in Y$. Formální kontext odvozený jednoduchým škálováním definujeme jako $\langle X, Z, J\rangle$, kde $Z = \bigcup_{y\in Y} Y_y$ a platí

$$\langle x, z\rangle \in J, \quad \text{právě když} \quad y(x) = v,\ \langle v, z\rangle \in I_y.$$

Existují další metody škálování vícehodnotových formálních kontextů, které lze nalézt v uvedených zdrojích.

1.4. Algoritmy pro výpočet konceptuálního svazu

Základním algoritmem pro výpočet všech konceptů daného formálního kontextu, který byl představený v článku Gantera [4], je algoritmus *NextClosure*. Tento algoritmus je ve své podstatě algoritmus pro výpočet pevných bodů libovolného uzávěrového operátoru. Jelikož z předchozího víme, že složením šipkových operátorů dostaneme uzávěrové operátory, lze tento algoritmus využít pro výpočet konceptů.

Pro jednodušší popis algoritmu *NextClosure* uvažujme $Y = \{1, \ldots, n\}$. Jinými slovy, atributy označíme navzájem různými přirozenými čísly a v důsledku získáme pevně dané uspořádání atributů.

Definice 1.21.
Pro libovolnou množinu atributů $B \subseteq Y$ definujeme

$$B \oplus i = ((B \cap \{1, \ldots, i-1\}) \cup \{i\})^{\downarrow\uparrow}.$$

Pro algoritmus *NextClosure* je zásadní definovat uspořádání na množinách atributů. Zavedeme proto lexikografické uspořádání množin atributů.

Definice 1.22.
Pro libovolné $A, B \subseteq Y,\ i \in \{1, \ldots, n\}$ definujeme

$$\begin{aligned} A <_i B \quad &\text{právě když} \quad i \in (B \setminus A),\ A \cap \{1, \ldots, i-1\} = B \cap \{1, \ldots, i-1\}, \\ A < B \quad &\text{právě když} \quad \exists i : A <_i B. \end{aligned}$$

Dostáváme uspořádání na množinách atributů, které využijeme v následující definici.

Definice 1.23.
Nechť $B \subseteq Y$. Nejmenší intent (vzhledem k „$<$“) větší než B je

$$B^+ = B \oplus i,$$

kde i je největší pro které platí $B <_i B \oplus i$.

Nyní máme zavedeno vše potřebné pro popis algoritmu *NextClosure*.

Algoritmus 1 NextClosure

```
1: procedure NextClosure(⟨X, Y, I⟩)
2:     B ← ∅^{↓↑};                                  ▷ Nejmenší intent.
3:     Result ← {B};
4:     while B ≠ Y do                               ▷ Dokud se nedostaneme k největšímu intentu.
5:         B ← B^+;
6:         Result ← Result ∪ B;
7:     end while
8:     return Result;
9: end procedure
```

Algoritmus *NextClosure* počítá množinu všech intentů (pevné body $^{\downarrow\uparrow}$). Množinu všech formálních konceptů lze dopočítat snadno. Stačí ke každému intentu B, přiřadit extent $B^{\downarrow}$. *NextClosure* může také počítat množinu všech extentů (pevné body $^{\uparrow\downarrow}$) a poté lze dopočítat koncepty výpočtem příslušných intentů. Důležité je si uvědomit, že *NextClosure* nám přímo nedává informaci o struktuře konceptuálního svazu (uspořádání konceptů).

Následující algoritmus, oproti algoritmu *NextClosure*, přímo poskytuje informaci o uspořádání konceptů. Jedná s o tzv. *Lindigův algoritmus - UpperNeighbor*.

Věta 1.24.
Nechť $\langle A, B\rangle \in \mathcal{B}(X, Y, I)$ je libovolný koncept mimo největší koncept. Potom $(A \cup \{x\})^{\uparrow\downarrow}$, kde $x \in (X \setminus A)$, je extent horního souseda $\langle A, B\rangle$, právě když

$$\forall z \in (A \cup \{x\})^{\uparrow\downarrow} \setminus A \quad \textit{platí} \quad (A \cup \{x\})^{\uparrow\downarrow} = (A \cup \{z\})^{\uparrow\downarrow}.$$

Předchozí věta je pro algoritmus *UpperNeighbor* stěžejní, protože udává návod, jak k danému konceptu vypočítat všechny jeho horní sousedy.

Algoritmus 2 UpperNeighbor

```
1: procedure UpperNeighbor(⟨A, B⟩ ∈ B(X, Y, I))
2:     candidates ← (X \ A);
3:     neighbors ← ∅;
4:     for x ∈ (X \ A) do
5:         B_1 ← (A ∪ {x})^↑;
6:         A_1 ← B_1^↓;
7:         if (candidates ∩ ((A_1 \ A) \ {x})) = ∅ then
8:             neighbors ← neighbors ∪ {⟨A_1, B_1⟩};
9:         else
10:            candidates ← candidates \ {x};
11:        end if
12:    end for
13:    return neighbors;
14: end procedure
```

Algoritmus *UpperNeighbor* počítá všechny horní sousedy daného konceptu. Tento fakt lze využít k výpočtu celého konceptuálního svazu i s jeho uspořádáním. Na začátku spustíme algoritmus se vstupem odpovídající nejmenšímu konceptu ($\langle \emptyset^{\uparrow\downarrow}, \emptyset^{\uparrow} \rangle$), čímž dostaneme všechny jeho horní sousedy. Na ty opět postupně aplikujeme stejný algoritmus, dokud se nedostaneme k největšímu konceptu ($\langle \emptyset^{\downarrow}, \emptyset^{\downarrow\uparrow} \rangle$).

1.5. Atributové implikace

Atributové implikace jsou výrazy, které popisují určitou závislost mezi atributy ve vstupních datech. Analogie atributových implikací se vyskytují v mnoha oblastech, například v oboru relačních databází jsou to *funkční závislosti*, v oboru dolování dat jsou to *asociační pravidla*. Jako první definujeme samotný pojem atributové implikace.

Definice 1.25 (atributová implikace)**.**
Nechť Y je neprázdná množina atributů. Atributová implikace je libovolný výraz

$$A \Rightarrow B,$$

kde $A \subseteq Y,\ B \subseteq Y$.

Nyní, když máme definovanou atributovou implikaci, je třeba definovat její *platnost*. Platnost atributové implikace se vždy vztahuje ke konkrétním datům. Potřebujeme nějakou základní sémantickou strukturu, ve které budeme vyhodnocovat pravdivost atributových implikací. Jako tuto základní sémantickou strukturu zvolíme řádky formálního kontextu. Na řádek formálního kontextu se můžeme dívat jako na charakterizaci jednoho objektu výčtem jeho atributů. Naším cílem je definovat platnost atributové implikace ve formálním kontextu.

Definice 1.26 (platnost atributové implikace)**.**
Atributová implikace $A \Rightarrow B$ nad Y je platná v množině $M \subseteq Y$, právě když

$$(A \subseteq M) \implies (B \subseteq M).$$

Atributová implikace $A \Rightarrow B$ nad Y je platná ve formálním kontextu $\langle X, Y, I \rangle$, právě když $A \Rightarrow B$ je platná pro každé $\{x\}^{\uparrow}$, kde $x \in X$.

Atributová implikace je platná ve formálním kontextu, právě když je platná ve všech řádcích tabulky příslušné danému formálnímu kontextu. Pokud ve formálním kontextu platí atributová implikace $A \Rightarrow B$, potom každý objekt, který má všechny atributy z A má i všechny atributy z B.

Poznámka 1.27.
Na atributové implikace se lze dívat jako na zkratky pro určité výrokové formule. Nechť $C = \langle X, Y, I \rangle$ je formální kontext, potom každé dvojici (x, y), $x \in X$, $y \in Y$ lze přiřadit výrokový symbol, jehož hodnota je dána $I(x, y)$ (pravda pokud $\langle x, y \rangle \in I$, jinak nepravda). Platnost atributové implikace $A \Rightarrow B$ ve formálním kontextu C je potom dána jako

$$\bigwedge_{x \in X} \left(\bigwedge_{y \in A} I(x, y) \Rightarrow \bigwedge_{z \in B} I(x, z) \right).$$

Nyní zavedeme analogii pojmů *teorie* a *model* z matematické logiky.

Definice 1.28 (teorie)**.**
Teorie nad Y je libovolná množina atributových implikací nad Y.

Definice 1.29 (model teorie)**.**
Model teorie T nad Y je libovolná množina $M \subseteq Y$ taková, že každá atributová implikace $A \Rightarrow B \in T$ je platná v M. Množinu všech modelů teorie T značíme $Mod(T)$.

Vybaveni pojmy teorie a modelu můžeme definovat *sémantické vyplývání.*

Definice 1.30 (sémantické vyplývání)**.**
Atributová implikace $A \Rightarrow B$ sémanticky plyne z teorie T (zn. $T \models A \Rightarrow B$), právě když $A \Rightarrow B$ je platná v každém modelu teorie T.

Některé atributové implikace plynou z jiných nebo jsou triviálně platné. Například atributová implikace $A \Rightarrow A$ je vždy triviálně splněna. Dalším příkladem je atributová implikace $A \Rightarrow C$, která určitě plyne z teorie $\{A \Rightarrow B,\ B \Rightarrow C\}$. Otázkou je, zda existuje systém pravidel, který by umožňoval ověřit, zda ze zadané teorie plyne daná atributová implikace. Takové systémy existují a uvedeme si zde nejznámější z nich, tzv. *Armstrongův axiomatický systém.*

Definice 1.31 (Armstrongův axiomatický systém)**.**
Armstrongův axiomatický systém se skládá z odvozovacích pravidel

$$(Ax) \quad \frac{}{A \cup B \Rightarrow A}, \qquad (Cut) \quad \frac{A \Rightarrow B, B \cup C \Rightarrow D}{A \cup C \Rightarrow D}.$$

Odvozovací pravidlo (Ax) umožňuje odvodit atributové implikace z prázdné množiny předpokladů. Pro takto odvozené atributové implikace vždy platí, že jejich pravá strana je podmnožinou levé. Druhé představené odvozovací pravidlo (Cut), neboli *pravidlo řezu*, již má neprázdnou množinu předpokladů. Pravidlo řezu proto, že skládáme dvě implikace, ze kterých „vyřízneme“ společnou část B.

Aby bylo možné odvozovat atributové implikace z představeného axiomatického systému, je třeba formálně zavést pojem *důkazu.* Za důkaz považujeme konečnou sekvenci atributových implikací, která splňuje určité vlastnosti.

Definice 1.32 (důkaz atributové implikace)**.**
Důkaz atributové implikace $A \Rightarrow B$ z teorie T je konečná sekvence atributových implikací $A_1 \Rightarrow B_1, \ldots, A_n \Rightarrow B_n$ splňující:

1. *$A_n \Rightarrow B_n$ je právě $A \Rightarrow B$,*
2. *$\forall i \in \{1, \ldots, n\}$:*
   - *buď $A_i \Rightarrow B_i \in T$,*
   - *nebo $A_i \Rightarrow B_i$ vznikne použitím odvozovacích pravidel (Ax), (Cut) na některé $A_j \Rightarrow B_j$, kde $j < i$.*

Pokud existuje důkaz atributové implikace $A \Rightarrow B$ z teorie T, potom tento fakt zapisujeme jako $T \vdash A \Rightarrow B$ a říkáme, že $A \Rightarrow B$ je dokazatelná z T pomocí odvozovacích pravidel (Ax) a (Cut).

Armstrongův axiomatický systém obsahuje pouze dvě základní odvozovací pravidla a dokazování pomocí nich může být zdlouhavé. Je tedy vhodné zavést dodatečná odvozovací pravidla, která dokazování usnadní. Tato dodatečná pravidla se nazývají *odvozená*.

Definice 1.33 (odvozené odvozovací pravidlo)**.**
Odvozovací pravidlo

$$\frac{A_1 \Rightarrow B_1, \ldots, A_n \Rightarrow B_n}{A \Rightarrow B}$$

nazveme odvozené z (Ax) a (Cut), pokud platí

$$\{A_1 \Rightarrow B_1, \ldots, A_n \Rightarrow B_n\} \vdash A \Rightarrow B.$$

Na *odvozená odvozovací pravidla* lze nahlížet jako na zkratky pro odvození pouze pomocí odvozovacích pravidel (Ax), (Cut) a lze je tedy používat v důkazech. To znamená, že náš odvozovací systém s nimi nebude silnější, ale bude přívětivější pro uživatele. Tím pádem také nemusíme měnit naši definici důkazu, protože každé použití odvozeného odvozovacího pravidla lze ekvivalentně nahradit sekvencí odvozovacích kroků používajících pouze (Ax), (Cut).

Věta 1.34.
Nechť $A, B, C, D \subseteq Y$, potom následující jsou odvozená odvozovací pravidla

$$\frac{}{A \Rightarrow A}, \qquad \frac{A \Rightarrow B}{A \cup C \Rightarrow B}, \qquad \frac{A \Rightarrow B, A \Rightarrow C}{A \Rightarrow B \cup C},$$

$$\frac{A \Rightarrow B \cup C}{A \Rightarrow B}, \qquad \frac{A \Rightarrow B, B \Rightarrow C}{A \Rightarrow C}.$$

Máme tedy způsob jak z teorie odvodit další atributové implikace, ale nevíme, zda jsou takto odvozené atributové implikace platné. Je třeba zavést pojem *korektnosti odvozovacího pravidla*. Korektnost znamená, že lze odvodit pouze platné atributové implikace. Jinými slovy, korektní odvozovací pravidlo je takové, pro které platí, pokud jsou všechny předpoklady (levá strana) platné v daných datech, musí platit i závěr (pravá strana).

Definice 1.35 (korektnost odvozovacího pravidla)**.**
Odvozovací pravidlo

$$\frac{A_1 \Rightarrow B_1, \ldots, A_n \Rightarrow B_n}{A \Rightarrow B}$$

je korektní, pokud platí $\{A_1 \Rightarrow B_1, \ldots, A_n \Rightarrow B_n\} \models A \Rightarrow B$.

Věta 1.36.
Odvozovací pravidla (Ax) *a* (Cut) *jsou korektní.*

Z předchozího poznatku o korektnosti odvozovacích pravidel (Ax), (Cut) dostáváme i korektnost od nich odvozených odvozovacích pravidel. To znamená, že cokoliv odvodíme pomocí předchozích odvozovacích pravidel je určitě platné. Nyní je potřeba prozkoumat druhou stránku věci, zda vše co je platné, lze představeným odvozovacím systémem dokázat. Tím získáme tzv. *úplnost* pro tento odvozovací systém.

Věta 1.37.
Armstrongův axiomatický systém je úplný, symbolicky:

$$T \vdash A \Rightarrow B, \;\; \textit{právě když} \;\; T \models A \Rightarrow B.$$

Armstrongův axiomatický systém je tedy úplný. Existují další známé systémy odvozovacích pravidel, které jsou ekvivalentní tomuto systému, pokud jsou také úplné.

Jak již bylo zmíněno, některé atributové implikace mohou plynout z nějaké teorie. Takové atributové implikace jsou v určitém smyslu nadbytečné (redundantní), protože nepřináší žádnou novou informaci. Zavedeme proto pojem *neredundantní teorie*.

Definice 1.38 (neredundantní teorie)**.**
Teorii T *nazveme neredundantní, pokud platí*

$$\forall A \Rightarrow B \in T : \; T \setminus \{A \Rightarrow B\} \not\models A \Rightarrow B.$$

Jinými slovy, teorii nazveme neredundantní, pokud žádná z jejich atributových implikací neplyne z ostatních. To znamená, že všechny atributové implikace v neredundantní teorii nesou určitou informaci, kterou bychom bez této atributové implikace neměli. Je nutné si uvědomit, že neredundantnost neznamená minimalitu vzhledem k počtu atributových implikací. Dalším důležitým pojmem je *úplná teorie*.

Definice 1.39 (úplná teorie)**.**
Nechť $\langle X, Y, I \rangle$ je formální kontext a T teorie nad Y. Teorii T nazveme úplnou v $\langle X, Y, I \rangle$, pokud platí

$$\forall A \Rightarrow B \in T: \ A \Rightarrow B \text{ je platná v } \langle X, Y, I \rangle, \ \text{právě když } \ T \vDash A \Rightarrow B.$$

Úplná teorie, vzhledem k danému formálnímu kontextu, v sobě nese úplnou informaci o platných atributových implikacích v daném formálním kontextu. Použitím pojmů neredundantní teorie a úplné teorie zavedeme pojem *neredundantní báze*.

Definice 1.40 (neredundantní báze)**.**
Nechť $\langle X, Y, I \rangle$ je formální kontext a T teorie nad Y. Teorii T nazveme neredundantní bází $\langle X, Y, I \rangle$, právě když T je neredundantní a úplná vzhledem k $\langle X, Y, I \rangle$.

Rozšířením pojmu neredundantní báze je *minimální neredundantní báze*, což je neredundantní báze, která obsahuje nejmenší možný počet atributových implikací.

2. Přehled vybraných přístupů k neúplným datům ve FKA

Formální konceptuální analýza, jak je představena v první kapitole, zpracovává úplná data. To znamená, že pokud chceme zpracovávat neúplná data, musíme k nim přistupovat zvláštním způsobem. Jedním z těchto způsobů je jejich předzpracování. Nejjednodušší možností je odstranění neúplných záznamů (objektů/atributů) a zachování pouze úplných dat, která umíme zpracovat. Odstraněním neúplných záznamů ale přijdeme o informace, které tato data nesla. Dalším způsobem předzpracování je zúplnění neúplných dat. To obnáší doplnění chybějících hodnot. Otázkou je, jak určit hodnoty, které máme doplnit. Můžeme například použít nějakou vlastní znalost nebo známou závislost v datech. Předzpracování neúplných dat není jedinou možností jak s takovými daty pracovat. Existují rozšíření metod FKA, která dokáží pracovat přímo s neúplnými daty bez jejich předzpracování. V této kapitole představím některé vybrané přístupy k neúplným datům ve FKA a to zejména metody zúplňování.

2.1. Základní pojmy

V první kapitole tohoto textu jsme definovali vícehodnotový formální kontext (1.18). Tuto definici je možné použít i nyní a to k definování neúplného formálního kontextu. Na *neúplný formální kontext* lze nahlížet jako na speciální případ již definovaného vícehodnotového formálního kontextu.

Definice 2.1 (neúplný formální kontext s relací)**.**
Neúplný formální kontext $\langle X, Y, V, I\rangle$ *je vícehodnotový formální kontext, kde* $V = \{\times, ?\}$.
Nechť $x \in X,\ y \in Y,\ v \in V$, *potom* $\langle x, y, v\rangle \in I$ *má význam:*

- x *má atribut* y, *pokud* $v = \times$,
- *není známo, zda* x *má atribut* y, *pokud* $v = ?$,
- *jinak* x *nemá atribut* y.

Neúplný formální kontext lze reprezentovat tabulkou, stejně jako v případě binárního formálního kontextu, ovšem v polích tabulky dovolíme nejen křížek a prázdné pole, ale i hodnotu „?“.

V některých situacích je výhodnější uvažovat místo relace $I \subseteq X \times Y \times V$ zobrazení $I : X \times Y \mapsto \{\times, -, ?\}$.

Definice 2.2 (neúplný formální kontext se zobrazením)**.**
Nechť $\langle X, Y, V, I\rangle$ je vícehodnotový formální kontext. Pokud I je zobrazení $I: X \times Y \mapsto \{\times, -, ?\}$, pak $I(x, y) = v$ má význam:

- *x má atribut y, pokud $v = \times$,*
- *x nemá atribut y, pokud $v = -$,*
- *není známo, zda x má atribut y, pokud $v = ?$.*

Obě představené definice neúplného formálního kontextu jsou ekvivalentní z hlediska obsahu informací. Neúplný formální kontext dle jedné definice, lze převést na odpovídající neúplný formální kontext dle druhé definice a to beze ztráty informace.

Spolu se zavedením neúplných formálních kontextů je vhodné zavést i další uspořádání, tentokrát na neúplných kontextech.

Definice 2.3 (uspořádání neúplných formálních kontextů)**.**
Nechť $C_1 = \langle X, Y, \{\times, -, ?\}, I_1\rangle$, $C_2 = \langle X, Y, \{\times, -, ?\}, I_2\rangle$ jsou neúplné formální kontexty. Jestliže C_2 vznikne z C_1 nahrazením nějakých otazníků jinými hodnotami $(\times, -)$, pak C_2 obsahuje více informací než C_1. Tento fakt zapisujeme jako $C_1 \leq C_2$.

Takto zavedené uspořádání neúplných formálních kontextů interpretujeme jako *uspořádání dle obsahu informací.*

Definice 2.4 (zúplnění neúplného formálního kontextu)**.**
Nechť $C_1 = \langle X, Y, \{\times, -, ?\}, I_1\rangle$ je neúplný formální kontext. Zúplnění neúplného formálního kontextu $C_1 = \langle X, Y, \{\times, -, ?\}, I_1\rangle$ je formální kontext C, který vznikne z C_1 nahrazením všech otazníků jinými hodnotami (nahrazení hodnotou „$-$“ lze interpretovat jako smazání dané n–tice z relace).

Zúplněním kontextu dostaneme maximální kontexty vzhledem k zavedenému informačnímu uspořádání a výsledný formální kontext neobsahuje žádné otazníky, takže na něj lze nahlížet jako na klasický binární formální kontext.

Dále zavedeme pojmy *jistý* a *možný intent (extent)* nad neúplným formálním kontextem $C = \langle X, Y, \{\times, -, ?\}, I\rangle$. *Jistý intent* množiny $A \subseteq X$ (zn. $A^{\Box}$) je množina všech atributů, které určitě mají (dovoluje pouze „$\times$“) všechny objekty z A. Naproti tomu *možný intent* množiny A (zn. $A^{\Diamond}$) je množina všech atributů, které možná mají (dovoluje i „?“) všechny objekty z A. Obdobně pro jistý a možný extent.

Definice 2.5 (jistý a možný extent/intent)**.**
Nechť $C = \langle X, Y, \{\times, -, ?\}, I\rangle$ je neúplný formální kontext. Pro množinu $A \subseteq X$ definujeme

$$A^{\Box} = \{y \in Y \mid (x, y, \times) \in I \text{ pro všechna } x \in A\},$$
$$A^{\Diamond} = \{y \in Y \mid (x, y, \times) \in I \text{ nebo } (x, y, ?) \in I \text{ pro všechna } x \in A\}.$$

Obdobně pro množinu $B \subseteq Y$ definujeme

$$\begin{aligned} B^{\square} &= \{x \in X \mid (x, y, \times) \in I \ \textit{ pro všechna } y \in B\}, \\ B^{\Diamond} &= \{x \in X \mid (x, y, \times) \in I \textit{ nebo } (x, y, ?) \in I \textit{ pro všechna } y \in B\}. \end{aligned}$$

Pokud platí $\{x\} = A,\ x \in X$ píšeme $x^{\square}$ místo $\{x\}^{\square}$ a také $x^{\Diamond}$ místo $\{x\}^{\Diamond}$. Obdobně pro $\{y\} = B,\ y \in Y$ píšeme $y^{\square}$ místo $\{y\}^{\square}$, $y^{\Diamond}$ místo $\{y\}^{\Diamond}$.

Pokud neúplný formální kontext $C = \langle X, Y, \{\times, -, ?\}, I\rangle$ neobsahuje žádný otazník, lze jej ztotožnit s klasickým binárním formálním kontextem $C = \langle X, Y, I\rangle$. V takovém případě platí $A^{\square} = A^{\Diamond} = A^{\uparrow_I}$ a také $B^{\square} = B^{\Diamond} = B^{\downarrow_I}$.

2.2. Zúplňování pomocí explorativní analýzy

Attribute exploration, neboli explorativní analýza, je metoda FKA pro získávání znalostí z neúplných dat. Konkrétně, explorativní analýza funguje následovně. Uživateli jsou postupně pokládány dotazy ohledně platnosti určitých atributových implikací. Pokud uživatel poskytne odpověď na všechny položené dotazy, výstupem explorativní analýzy je báze všech platných atributových implikací v datech a navíc, ke každé neplatné atributové implikaci alespoň jeden protipříklad. Může se ale stát, že uživatel nezná odpověď na všechny položené dotazy. V takovém případě je výstupem množina všech platných atributových implikací, množina atributových implikací, které mohou být platné, a množina vygenerovaných protipříkladů atributových implikací, u kterých uživatel neznal odpověď na dotaz ohledně jejich platnosti. Na základě výsledných atributových implikací je možné neúplná data zúplnit.

Algoritmů pro explorativní analýzu existuje několik typů. Liší se zejména tím, zda dovolují uživateli odpovědět „nevím“ na položený dotaz. Jinými slovy, jeden typ algoritmů pracuje s neúplnou znalostí platných atributových implikací, kdežto druhý typ pracuje pouze s úplnou znalostí platných atributových implikací. Zde bude uveden typ pracující s neúplnou znalostí platných atributových implikací. Hlavními zdroji pro tuto část byly články od Burmeistera, Holzera [5],[6],[7] a Obiedkova [8]. Jako definici neúplného formálního kontextu budeme používat definici 2.2.

2.2.1. Rozšíření atributových implikací

V první kapitole jsme zavedli pojem atributové implikace (1.25) a související pojem platnosti atributové implikace ve formálním kontextu (1.26). Nyní potřebujeme zavést analogii pojmu platnosti atributové implikace, tentokrát v neúplném formálním kontextu. Dále také zavedeme nový pojem *splnitelnosti*.

Definice 2.6 (platnost a splnitelnost atributových implikací)**.**
Nechť $C = \langle X, Y, \{\times, -, ?\}, I\rangle$ *je neúplný formální kontext. Atributová implikace* $A \Rightarrow B$ *nad* Y *je*

- *platná, právě když je platná ve všech zúplněních* C*, tzn.*

$$\forall x \in X : \ (A \subseteq x^{\Diamond}) \to (B \setminus A \subseteq x^{\Box});$$

- *splnitelná, právě když je platná alespoň v jednom zúplnění* C*, tzn.*

$$\forall x \in X : \ (A \subseteq x^{\Box}) \to (B \setminus A \subseteq x^{\Diamond}).$$

Z platnosti atributové implikace v neúplném formálním kontextu určitě plyne její splnitelnost. Každá platná atributová implikace v daném neúplném formálním kontextu je splnitelná. Naopak to ovšem neplatí.

2.2.2. Aplikace Kleeneho tříhodnotové logiky

Jak víme z první kapitoly (pozn. 1.27), atributové implikace lze chápat jako zkratky pro zápis určitých formulí výrokové logiky. Navíc, atributová implikace je platná v daném formálním kontextu, právě když je platná odpovídající formule výrokové logiky. Je proto vhodné rozšířit tento princip i pro neúplné kontexty. Jelikož máme tři možné hodnoty atributů místo dvou, nabízí se možnost využít tříhodnotové logiky. Logika, která má velmi blízko k představené sémantické definici platnosti a splnitelnosti atributových implikací je *Kleeneho tříhodnotová logika*. Pravdivostní tabulky logických spojek této logiky lze nalézt v tabulce 1.

Vyhodnocení formulí v Kleeneho tříhodnotové logice ovšem plně neodpovídá definované platnosti a splnitelnosti atributových implikací. Například atributová implikace $\{y\} \Rightarrow \{y\}$ je vždy triviálně platná, ale pokud je hodnota y pro nějaký objekt rovna „?“, potom pravdivostní hodnota formule odpovídající dané atributové implikaci je „?“. Jedná se o důsledek rozdílného významu hodnoty „?“ v neúplných formálních kontextech a Kleeneho tříhodnotové logice. V neúplném formálním kontextu je význam hodnoty „?“ interpretován jako nedostatek informací a jedná se pouze o zástupce pro jednu z hodnot „−“ nebo „×“. Naproti ostatním hodnotám, hodnota „?“ nemá žádný vztah k reálné doméně atributů vstupních dat.

$\wedge$	-	?	×
-	-	-	-
?	-	?	?
×	-	?	×

$\Rightarrow$	-	?	×
-	×	×	×
?	?	?	×
×	-	?	×

$\neg$	
-	×
?	?
×	-

Tabulka 1: Pravdivostní tabulky spojek Kleeneho tříhodnotové logiky.

V případě, kdy výroková formule odpovídající atributové implikaci $A \Rightarrow B$ obsahuje každou výrokovou proměnnou nejvýše jedenkrát (platí $A \cap B = \emptyset$), potom Kleeneho tříhodnotová logika odpovídá představené sémantické definici platnosti a splnitelnosti atributových implikací. Jelikož atributová implikace $A \Rightarrow B$ platí, právě když platí $A \Rightarrow (B \setminus A)$, můžeme předchozí použít pro výpočet platnosti libovolné atributové implikace v neúplném formálním kontextu. Kleeneho tříhodnotová logika ovšem nedostačuje pro výpočet platnosti libovolné množiny atributových implikací, protože výroková formule odpovídající množině atributových implikací může nevyhnutelně obsahovat některou výrokovou proměnnou vícekrát.

2.2.3. Aplikace modální logiky

Ve článku Obiedkova [8] je představena modální logika navržená za účelem vyhodnocování libovolných formulí nad atributy daného neúplného formálního kontextu.

Modální logika rozšiřuje predikátovou logiku o operátory vyjadřující možnost a nutnost. Je v ní tedy možné formalizovat výroky typu „Je možné, že dnes bude pršet.“. Sémantika modální logiky je založena na tzv. *možných světech*, přičemž možné světy mohou být jeden z druhého dosažitelné.

V našem případě definujeme možné světy jako všechny neúplné kontexty nad danou množinou atributů Y a také všechny podmnožiny Y (reprezentují možné intenty objektů). Z neúplného formálního kontextu $C = \langle X, Y, \{\times, -, ?\}, I\rangle$ jsou dosažitelná všechna jeho zúplnění a všechny intenty objektů těchto zúplnění. Z množiny atributů není dosažitelné nic dalšího. Interpretace třetí pravdivostní hodnoty je dána jako „nesmysl“, což ovlivňuje význam modálních operátorů. Výrok „Je nutné, že ϕ.“ je v možném světě w pravdivý, jestliže je pravdivý v každém světe dosažitelném z w, ve kterém má smysl a jen pokud takový existuje. Dále tento výrok nemá smysl ve světě w, pokud nemá smysl v žádném z dosažitelných světů z w nebo když z něj není žádný svět dosažitelný. Množina atributových implikací A je platná, tehdy a jen tehdy, pokud je pravdivý výrok „Je nutné, že ϕ.“, kde ϕ je formule odpovídající A, což je konjunkce formulí odpovídajících jednotlivým atributovým implikacím, jak jsou popsány v poznámce 1.27. Splnitelnost A je ekvivalentní pravdivostní hodnotě výroku „Je možné, že je nutné ϕ.“, kde ϕ je stejná formule jako v předchozím případě.

Představená modální logika již netrpí nedostatky zmíněnými u Kleeneho tříhodnotové logiky. Pravdivostní tabulky základních logických spojek představené modální logiky lze nalézt v tabulce 2. Další logické spojky se definují jako v klasické výrokové logice.

$\wedge$	-	?	$\times$
-	-	?	-
?	?	?	?
$\times$	-	?	$\times$

$\Rightarrow$	-	?	$\times$
-	$\times$	?	$\times$
?	?	?	?
$\times$	-	?	$\times$

$\neg$	
-	$\times$
?	?
$\times$	-

Tabulka 2: Pravdivostní tabulky spojek modální logiky pro formule nad atributy v neúplném formálním kontextu.

V pravdivostních tabulkách logických spojek představené modální logiky je třetí pravdivostní hodnota i na místech, kde Kleeneho tříhodnotová logika udává jednu ze dvou klasických pravdivostních hodnot. To je dáno právě interpretací třetí pravdivostní hodnoty jako „nesmysl“. Jinými slovy, pokud výrok obsahuje nesmyslnou část, tak je vždy celý nesmyslný.

2.2.4. Struktura algoritmu explorativní analýzy

Nyní již můžeme představit základní strukturu algoritmu explorativní analýzy, jak byla původně představena nad Kleeneho tříhodnotovou logikou. Zájemcům o hlubší studium této metody doporučuji zejména články Holzera [5],[6].

1. Uživatel na počátku zadá neúplný formální kontext (může obsahovat otazníky), který označme jako aktuální. Dále může zadat nějakou vlastní znalost ve formě atributových implikací, tzv. *background knowledge*, o univerzu vstupních dat. Množina přijatých atributových implikací se inicializuje na prázdnou množinu (na počátku není přijatá žádná).

2. V j-tém průběžném kroku označme aktuální kontext jako C_j. Algoritmus zvolí atributovou implikaci $A \Rightarrow B$, která je splnitelná. Přičemž A je nejmenší (vzhledem k „$\subseteq$“) množina, ve které jsou platné všechny doposud přijaté atributové implikace a $B = \{y \in Y \mid A \Rightarrow \{y\}$ je splnitelná v $C_j\}$. Pokud taková atributová implikace neexistuje, algoritmus končí. Množina B tedy obsahuje všechny atributy y, pro které je atributová implikace $A \Rightarrow \{y\}$ splnitelná v aktuálním kontextu. Pokud $A \Rightarrow B$ je odvoditelná (pomocí Armstrongova axiomatického systému) z množiny doposud přijatých atributových implikací a zadané vlastní znalosti, je přijata automaticky. Jinak je uživateli položen dotaz, zda je atributová implikace $A \Rightarrow B$ platná v daném univerzu.

   - Pokud uživatel odpoví „ano“, potom je atributová implikace $A \Rightarrow B$ přidána do množiny přijatých atributových implikací.
   - Pokud uživatel odpoví „ne“, musí zadat objekt (řádek tabulky), který je protipříkladem atributové implikace $A \Rightarrow B$. Nový řádek může ob-

sahovat opět otazníky. Objekt odpovídající takovému řádku je přidán k aktuálnímu kontextu.

- Pokud uživatel odpoví „nevím“, potom musí přesně specifikovat, pro které $b \in B$ je platnost atributové implikace $A \Rightarrow \{b\}$ neznámá. Označme $Z_j = \{b \in B \mid \text{platnost } A \Rightarrow \{b\} \text{ je neznámá}\}$. Pro každý atribut $b \in Z_j$ algoritmus vytvoří imaginární objekt, který je nejmenší (vzhledem k informačnímu uspořádání) protipříklad pro atributovou implikaci $A \Rightarrow \{b\}$. Takový objekt má všechny atributy z A, nemá atribut b a ostatní atributy jsou neznámé. Pokud $B \setminus Z_j \neq A$, pak atributová implikace $A \Rightarrow B \setminus Z_j$ je přidána do množiny přijatých atributových implikací.

Po dokončení algoritmu explorativní analýzy může být třeba dodatečných akcí. Jednou takovou akcí je například odstranění nadbytečných imaginárních objektů. Může totiž nastat situace, že atributová implikace $A \Rightarrow \{b\}$, o které se neví zda je platná, plyne z výsledné množiny přijatých atributových implikací. V takovém případě je nutné odstranit imaginární objekt, který byl přidán jako protipříklad pro tuto atributovou implikaci. Dále lze odebrat nadbytečné imaginární protipříklady pro atributové implikace, pro které ve výsledném kontextu existuje reálný protipříklad.

Otázkou zůstává, jak zúplnit neúplný formální kontext na základě výsledných atributových implikací. Označme T množinu atributových implikací, které mají být platné ve zúplnění neúplného kontextu $C = \langle X, Y, \{\times, -, ?\}, I\rangle$. Je možné nahradit hodnotu „?“ příslušnou objektu $x \in X$ a atributu $y \in Y$ hodnotou

- „$\times$“ - pokud existuje $A \Rightarrow B \in T$ taková, že $A \subseteq x^{\square}$ a $y \in B$;
- „$-$“ - pokud existuje $A \Rightarrow B \in T$ taková, že $x \in A,\ A \setminus \{x\} \subseteq x^{\square}$ a $B \nsubseteq x^{\Diamond}$.

2.3. Zúplňování pomocí maticové faktorizace

Metoda zúplňování neúplného formálního kontextu pomocí maticové faktorizace byla představena v článku Piskové, Pera, Horvátha a Krajči [9], ze kterého jsem při zpracování této části vycházel.

Nejdříve připomeneme pojmy *matice daného typu* a *množina všech matic daného typu*, zejména k ujasnění značení.

Definice 2.7 (matice)**.**
Nechť T je číselné těleso a $m, n \in \mathbb{N}$. Pak zobrazení

$$A: \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \mapsto T$$

se nazývá matice typu $m \times n$, kterou zapisujeme jako obdélníkové schéma

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix},$$

kde $a_{ij} = A(i, j)$. Množinu všech matic typu $m \times n$ nad číselným tělesem T značíme $\mathcal{M}_{m \times n}(T)$.

2.3.1. Faktorizace matice

Maticová faktorizace je často používaná metoda v oblasti dolování dat. Pomocí ní lze rozložit matici na menší matice tzv. *faktory*, které při zpětném složení dávají aproximaci původní matice. Pro faktorizaci matic existuje několik metod. Jednou z nich je *stochastická gradientní faktorizace*. Výstupem této metody pro matici $C \in \mathcal{M}_{|X| \times |Y|}(\mathbb{R})$ je dvojice matic $W \in \mathcal{M}_{|X| \times K}(\mathbb{R})$, $H \in \mathcal{M}_{|Y| \times K}(\mathbb{R})$ taková, že $WH^T = C'$ aproximuje matici C, značeno $C \approx C' = WH^T$. Číslo K udává počet faktorů. Nechť $x_{ij} \in C$ je neznámá hodnota. Odhad x_{ij} je x'_{ij}, který je dán jako

$$x'_{ij} = (WH^T)_{ij}.$$

Jinými slovy, odhad neznáme hodnoty v C je hodnota na dané pozici ve vypočítané aproximaci C'. Nyní je potřeba vyřešit, jak získat takové C', které dobře odhaduje neznáme hodnoty. Za tímto účelem se C rozdělí do dvou komplementárních částí C^{train}, C^{test}, tzn. $C^{train} \subseteq C$, $C^{test} = C \setminus C^{train}$. Množinu C^{train} označme jako *tréninkovou množinu* a C^{test} jako *testovací množinu*. Aproximace C' je určena pouze ze známých hodnot z C^{train} a přesnost této aproximace je dána pomocí *směrodatné odchylky* nad maticí C^{test}, která je dána jako

$$s(C^{test}) = \sqrt{\frac{\sum_{x_{ij} \in C^{test}} (x_{ij} - x'_{ij})^2}{|C^{test}|}},$$

kde x_{ij} jsou pouze známe hodnoty z C^{test}. Započítáváme jen známé hodnoty, protože u neznámých hodnot nemáme jak spočítat odchylku odhadu. Cílem je nalézt takové C' (respektive W, H), pro které je směrodatná odchylka minimální. Za tímto účelem se použije stochastická gradientní metoda, která hledá minimum funkce

$$\sum_{x_{ij} \in C^{train}} (x_{ij} - x'_{ij})^2 + \lambda(||w_i||^2 + ||h_j||^2),$$

kde w_i je i-tý řádek matice W, h_j je j-tý řádek matice H, λ je parametr, který zabraňuje tzv. *přeučení*. Přeučení je jev, který nastává, když jsme schopni velmi přesně odhadovat data v tréninkové množině, ale odhad nad testovací množinou selhává.

2.3.2. Hledání vhodné faktorizace

Ve fázi hledání vhodného rozkladu matice C jsou nejprve matice W, H inicializovány na náhodné hodnoty. Dále dle vypočítané odchylky se inkrementálně upravují jejich prvky. Tyto úpravy ovlivňuje další volitelný parametr. Celkově tedy existuje několik parametrů faktorizace, které je třeba zvolit.

Při snaze získat vhodné W, H může nastat situace, kdy prvky C^{train} a C^{test} nesdílí žádné podobné rysy. K zabránění tomuto jevu lze použít tzv. *n-fold cross validaci*, kde data rozdělujeme do n stejně velkých částí. Zvolíme sadu parametrů faktorizace a pro každou zvolenou část vypočítáme pomocí předchozí metody faktorizaci jejího doplňku a spočítáme směrodatnou odchylku pro tuto část. Výsledná směrodatná odchylka pro C' s danými parametry učení je potom průměr všech směrodatných odchylek jednotlivých částí. Po vyzkoušení několika sad parametrů faktorizace vybereme tu s nejmenší výslednou směrodatnou odchylkou.

2.3.3. Experimenty

V článku [9] lze nalézt tři experimenty s tímto druhem zúplňování. Použitá data byla vybrána z *UCI machine learning repository* [10], u kterých byly provedeny testy s různým procentem neznámých hodnot (10 %, 20 %, 30 %). Průměrná úspěšnost v těchto experimentech byla okolo 55 %. Zajímavé ale je, že u dvou ze tří provedených experimentů byla úspěšnost téměř identická, nezávisle na procentu neznámých dat. Jelikož se jedná o experimentální metodu, autoři článku tento fakt dále nijak nekomentovali.

2.4. Ostatní přístupy

Problematika neúplných dat se stále rozvíjí a již existuje několik metod pro jejich zpracování pomocí FKA. V předchozích částech jsem podrobněji rozebral dva vybrané přístupy. V této části několika málo větami shrnu obsah dalších článků, které jsem v této oblasti studoval.

V článku Lia a Yaa [11] je rozvedena idea škálování neúplného formálního kontextu. Víme, že neúplný formální kontext je vlastně speciální případ vícehodnotového kontextu. V první kapitole je popsán proces škálování neúplného kontextu, který vede k získání binárního formálního kontextu. V tomto článku se autoři zabývají závislostmi mezi jednotlivými atributy a jejich vlivem na volbu škál a způsob škálování neúplných formálních kontextů.

Jedním z přístupů, který není založený na předzpracování dat, ale na přizpůsobení/rozšíření metod FKA pro neúplná data je představen v článku Krupky a Laštovičky [12]. V něm je představena metoda konstrukce konceptuálního svazu pro neúplné formální kontexty a to takovým způsobem, že výsledný konceptuální svaz v sobě nese informace o konceptuálních svazech všech možných zúplnění. Tato teorie má jako svůj základ vícehodnotovou logiku, kde mezilehlé stupně pravdivosti představují buď proměnné nebo hodnoty vypočítané z těchto proměnných pomocí logických operací.

Další přístup ke zúplňování neúplných formálních kontextů je představen v článku Othmana a Yahia [13]. Představená metoda je založena na asociačních pravidlech a řeší problém konfliktu pravidel při jejich použití pro zúplňování. Konkrétně, je zde představena míra pro asociační pravidla, dle které se vybere vhodné asociační pravidlo, na základě kterého se doplní neznámá hodnota. Představená metoda ale nezaručuje, že po jejím provedení dostaneme úplný formální kontext.

3. Vlastní výsledky

Při studiu formální konceptuální analýzy nad neúplnými daty jsem se zároveň snažil přijít s vlastními poznatky z této oblasti. Jako první jsem zkoumal experimentální metodu zúplňování neúplného formální kontextu, který obsahuje právě jeden otazník. Jedná se tedy o problém predikce právě jedné neznámé hodnoty. Tomuto je věnována první část této kapitoly. Ukázalo se, že tento problém vede na jiný, obecnější problém, který je neméně zajímavý. Jedná se o zkoumání změn v konceptuálním svazu po odebrání incidence z příslušného formálního kontextu. Právě tomuto tématu se věnuje druhá část.

V celé této kapitole je použito následující značení:

- $C = \langle X, Y, \{\times, -, ?\}, K\rangle$ - neúplný formální kontext vstupních dat, který obsahuje právě jeden otazník příslušný objektu x_0 a atributu y_0.
- $C(I) = \langle X, Y, \{\times, -, ?\}, I\rangle$ - úplný formální kontext, který vznikne zúplněním C nahrazením otazníku hodnotou „$\times$“.
- $C(J) = \langle X, Y, \{\times, -, ?\}, J\rangle$ - úplný formální kontext, který vznikne zúplněním C nahrazením otazníku hodnotou „$-$“.
- $\mathcal{B},\ \mathcal{B}(I),\ \mathcal{B}(J)$ - příslušné konceptuální svazy.

Formální kontexty $C(I),\ C(J)$ jsou úplné a liší se pouze v relaci incidence. Konkrétně, jedna vznikne z druhé odebráním/přidáním právě jedné incidence $\langle x_0, y_0\rangle$. Jelikož jsou $C(I),\ C(J)$ úplné, lze je ztotožnit s klasickými binárními formální kontexty $C(I) = \langle X, Y, I\rangle,\ C(J) = \langle X, Y, J\rangle$.

3.1. Zúplňování pomocí počítání konceptů

Zúplnění neúplného formálního kontextu lze provést několika způsoby, jak je možné vidět v předchozí kapitole. Jedním z prvotních cílů této práce bylo prozkoumat experimentální přístup ke zúplňování, který je založen na počítání konceptů. Idea této metody vzešla z experimentů původně provedených pro jinou práci. Tyto experimenty byly provedeny na několika vybraných formálních kontextech. Výsledkem byla zajímavá úspěšnost predikce neznámé hodnoty a proto jsem se této metodě věnoval.

Myšlenka zúplnění neúplného formálního kontextu pomocí počítání konceptů je následující:

- Nechť vstupní data (neúplný formální kontext C) obsahují právě jednu neznámou hodnotu (jeden otazník).

- Zúplníme C nahrazením otazníku hodnotou, která přísluší konceptuálnímu svazu s menším počtem konceptů. Jinými slovy, pokud platí $|\mathcal{B}(J)| < |\mathcal{B}(I)|$, hodnota pro zúplnění je „$-$“. Oproti tomu, pokud platí $|\mathcal{B}(I)| < |\mathcal{B}(J)|$, hodnota pro zúplnění je „$\times$“. Pokud $|\mathcal{B}(I)| = |\mathcal{B}(J)|$, nelze na základě tohoto postupu rozhodnout.

Vyvstává přirozená otázka, proč by tento přístup měl fungovat. Z první kapitoly víme, že formální konceptuální analýza je metoda pro analýzu dat. Odhaluje v datech shluky (formální koncepty), které mnohdy mají přirozenou interpretaci (představují známý pojem). Dle pravidla „správná odpověď je vždy ta nejjednodušší“ je správné to zúplnění, které obsahuje méně konceptů (je jednodušší). To ale také znamená, že tato metoda se vždy snaží zjednodušit vzory ve vstupních datech a potlačuje prvky vymykající se již přítomným vzorům. Jinými slovy, tato metoda se nesnaží přinést novou informaci do vstupních dat, ale naopak se snaží, aby tato informace zapadla do již přítomné struktury dat.

Jelikož se jedná o experimentální metodu, bylo nutné provést několik experimentů. Aby tyto experimenty bylo možné provést, zaměřil jsem se prvotně na hlubší zkoumání tohoto problému a vytvoření základního algoritmu pro tuto metodu zúplňování.

Naivní algoritmus pro zúplnění pomocí počítání konceptů spočítá koncepty v obou zúplněních a porovná vypočtené hodnoty. Výpočet konceptuálních svazů $\mathcal{B}(I),\ \mathcal{B}(J)$ lze provést například některým z algoritmů, které jsou uvedeny v první kapitole. Zároveň je vhodné se zamyslet, zda tento problém nelze řešit efektivněji, protože výpočet konceptuálního svazu je výpočetně velmi náročný. To mě přivedlo k problému odvození počtu konceptů jednoho konceptuálního svazu ze druhého. Došel jsem k závěru, že pro tento účel je výhodnější vycházet ze znalosti $\mathcal{B}(I)$. Ukázalo se, že pro výpočet rozdílu počtu konceptů ani není nutné počítat celý konceptuální svaz $\mathcal{B}(I)$, ale stačí vypočítat jeho určitou část.

3.1.1. Základní pojmy a jejich vlastnosti

Nejprve zavedeme vhodné pojmy k uchopení představeného problému. Jako první objasníme vztah, který mezi sebou mají šipkové operátory $^{\uparrow_I}, ^{\uparrow_J}, ^{\downarrow_I}, ^{\downarrow_J}$, příslušné oběma možným zúplněním. Následující věta je velmi často využívána ve zbytku této kapitoly a je vhodné ji dobře prostudovat.

Věta 3.1 (vztah šipkových operátorů $^{\uparrow_I}, ^{\uparrow_J}, ^{\downarrow_I}, ^{\downarrow_J}$)**.**
Pro libovolnou $A \subseteq X$ a $B \subseteq Y$ platí

$$A^{\uparrow_J} = \begin{cases} A^{\uparrow_I} & \text{jestliže } x_0 \notin A, \\ A^{\uparrow_I} \setminus \{y_0\} & \text{jestliže } x_0 \in A, \end{cases} \qquad B^{\downarrow_J} = \begin{cases} B^{\downarrow_I} & \text{jestliže } y_0 \notin B, \\ B^{\downarrow_I} \setminus \{x_0\} & \text{jestliže } y_0 \in B. \end{cases}$$

Konkrétně, $A^{\uparrow_J} \subseteq A^{\uparrow_I}$ a $B^{\downarrow_J} \subseteq B^{\downarrow_I}$.

Důkaz. Ihned ze zavedení $C(I),\ C(J)$ a z faktu, že $J = I \setminus \{\langle x_0, y_0\rangle\}$. □

Při zkoumání $\mathcal{B}(I)$ a $\mathcal{B}(J)$ si lze všimnout, že mohou sdílet některé formální koncepty. Zavedeme tedy pojem *stabilní koncept*. Jako stabilní označíme koncepty patřící do konceptuálních svazů obou zúplnění $C(I),\ C(J)$. To znamená, že při výpočtu $\mathcal{B}(I),\ \mathcal{B}(J)$ není nutné tyto koncepty počítat dvakrát.

Definice 3.2 (stabilní koncept)**.**
Koncept $c \in \mathcal{B}(I) \cup \mathcal{B}(J)$ nazveme stabilním, právě když $c \in \mathcal{B}(I) \cap \mathcal{B}(J)$.
Koncept $c \in \mathcal{B}(I) \cup \mathcal{B}(J)$ nazveme nestabilním, právě když není stabilní.

Pojem stability rozděluje koncepty $\mathcal{B}(I),\ \mathcal{B}(J)$ na dvě disjunktní množiny, které tvoří pokrytí všech konceptů v každém z těchto konceptuálních svazů. Každý koncept je buď stabilní nebo nestabilní. Při výpočtu $\mathcal{B}(I),\ \mathcal{B}(J)$ není nutné stabilní koncepty počítat vícekrát. Naopak nestabilní koncepty jsou právě ty, které patří právě do jednoho z $\mathcal{B}(I),\ \mathcal{B}(J)$. To znamená, že se můžeme zaměřit pouze na určitou podmnožinu konceptů a zabývat se vlastnostmi jen těchto konceptů. Za tímto účelem definujeme čtveřici operátorů, které nám v tomto pomohou.

Definice 3.3 (operátory ${}^{\square}, {}_{\square}, {}^{\boxtimes}, {}_{\boxtimes}$)**.**
Pro koncept $c = \langle A, B\rangle \in \mathcal{B}(I),\ d = \langle C, D\rangle \in \mathcal{B}(J)$ definujeme

$$c^{\square} = \langle A^{\square}, B^{\square}\rangle = \langle A^{\uparrow_J\downarrow_J}, A^{\uparrow_J}\rangle, \qquad c_{\square} = \langle A_{\square}, B_{\square}\rangle = \langle B^{\downarrow_J}, B^{\downarrow_J\uparrow_J}\rangle,$$
$$d^{\boxtimes} = \langle C^{\boxtimes}, D^{\boxtimes}\rangle = \langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle, \qquad d_{\boxtimes} = \langle C_{\boxtimes}, D_{\boxtimes}\rangle = \langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle.$$

Z předchozí definice je jasné, že pro libovolné $c \in \mathcal{B}(I)$ platí $c^{\square}, c_{\square} \in \mathcal{B}(J)$. Podobně, pro libovolné $d \in \mathcal{B}(J)$ platí $d^{\boxtimes}, d_{\boxtimes} \in \mathcal{B}(I)$. Účelem operátorů z předchozí definice je zachytit určité spojení mezi koncepty z $\mathcal{B}(I)$ a $\mathcal{B}(J)$, které s výhodou využijeme při tvorbě cílového algoritmu.

Ukázalo se, že dvojice $\langle {}^{\square}, {}^{\boxtimes}\rangle$, $\langle {}_{\square}, {}_{\boxtimes}\rangle$ mají vlastnosti podobné Galoisovým konexím, což je vítané, ale nejedná se přímo o Galoisovy konexe. Netvoří antitonní Galoisovy konexe, protože všechna zobrazení jsou izotonní. Netvoří ani izotonní Galoisovy konexe, protože v takovém případě složené operátory tvoří uzávěrový a vnitřkový operátor. V našem případě jsou oba složené operátory stejného typu.

Teorém 3.4 (vlastnosti operátorů ${}^{\square}, {}_{\square}, {}^{\boxtimes}, {}_{\boxtimes}$)**.**
Nechť $c \in \mathcal{B}(I),\ d \in \mathcal{B}(J)$. Zobrazení $c \mapsto c^{\square}$, $c \mapsto c_{\square}$ a $d \mapsto d^{\boxtimes}$, $d \mapsto d_{\boxtimes}$ jsou izotonní a splňují

$$c \le c^{\square\boxtimes}, \qquad d \le d^{\boxtimes\square}, \qquad c^{\square\boxtimes\square} = c^{\square}, \qquad d^{\boxtimes\square\boxtimes} = d^{\boxtimes},$$
$$c \ge c_{\square\boxtimes}, \qquad d \ge d_{\boxtimes\square}, \qquad c_{\square\boxtimes\square} = c_{\square}, \qquad d_{\boxtimes\square\boxtimes} = d_{\boxtimes}.$$

Důkaz. Dokážeme, že předchozí platí pro zobrazení $c \mapsto c^{\square}$ a $d \mapsto d^{\boxtimes}$. Důkaz pro $c \mapsto c_{\square}$ a $d \mapsto d_{\boxtimes}$ je duální.

Nechť $c_1 = \langle A_1, B_1\rangle,\ c_2 = \langle A_2, B_2\rangle \in \mathcal{B}(I),\ c_1 \le c_2$. Potom $A_1 \subseteq A_2$ a $A_1^{\uparrow_J\downarrow_J} \subseteq A_2^{\uparrow_J\downarrow_J}$, takže ${c_1}^{\square} \le {c_2}^{\square}$.

Nechť $d_1 = \langle C_1, D_1\rangle,\ d_2 = \langle C_2, D_2\rangle \in \mathcal{B}(J),\ d_1 \le d_2$. Potom $D_2 \subseteq D_1$ a $D_2^{\downarrow_I\uparrow_I} \subseteq D_1^{\downarrow_I\uparrow_I}$, takže ${d_1}^{\boxtimes} \le {d_2}^{\boxtimes}$.

Dále nechť $c = \langle A, B\rangle \in \mathcal{B}(I)$. Z věty 3.1 máme $A^{\uparrow_J} \subseteq A^{\uparrow_I}$. Z toho, že $A = A^{\uparrow_I\downarrow_I} \subseteq A^{\uparrow_J\downarrow_I}$ dostáváme $c \leq c^{\square\boxtimes}$. Obdobně, pro $d = \langle C, D\rangle \in \mathcal{B}(J)$, $D^{\downarrow_J} \subseteq D^{\downarrow_I}$, takže $D^{\downarrow_I\uparrow_J} \subseteq D^{\downarrow_J\uparrow_J} = D$ a tedy $d \leq d^{\boxtimes\square}$.

K dokázání platnosti $c^{\square\boxtimes\square} = c^{\square}$ stačí dokázat, že pro extent A konceptu c platí $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J}$. Dle věty 3.1 existují dvě možnosti: buď $A^{\uparrow_J} = A^{\uparrow_I}$ nebo $A^{\uparrow_J} = A^{\uparrow_I} \setminus \{y_0\}$. V prvním případě $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J}$ platí triviálně z vlastností Galoisových konexí. Ve druhém případě $A^{\uparrow_J\downarrow_I} = A^{\uparrow_J\downarrow_J}$ (z věty 3.1, protože $y_0 \notin A^{\uparrow_J}$) a $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J\downarrow_J\uparrow_J} = A^{\uparrow_J}$. Rovnost $d^{\boxtimes\square\boxtimes} = d^{\boxtimes}$ se dokazuje analogicky. □

Přímým důsledkem předchozího jsou vlastnosti zobrazení, která vzniknou složením $c \mapsto c^{\square},\ d \mapsto d^{\boxtimes}$ a $c \mapsto c_{\square},\ d \mapsto d_{\boxtimes}$. Jednotlivá složená zobrazení vždy tvoří uzávěrový nebo vnitřkový operátor.

Důsledek 3.5 (složené operátory $^{\square\boxtimes}, {}_{\square\boxtimes}$)**.**
Složená zobrazení $c \mapsto c^{\square\boxtimes}$, $d \mapsto d^{\boxtimes\square}$ jsou uzávěrové operátory.
Složená zobrazení $c \mapsto c_{\square\boxtimes}$, $d \mapsto d_{\boxtimes\square}$ jsou vnitřkové operátory.

Při zkoumání zavedených operátorů si lze všimnout několika zjevných vlastností, které jsou užitečné jednak pro zjednodušení jejich výpočtu, ale také pro dokazování dalších netriviálních vlastností.

Věta 3.6.
Pokud pro koncept $c \in \mathcal{B}(I)$ platí $c = c^{\square}$, potom $c = c^{\square\boxtimes}$.
Pokud pro koncept $c \in \mathcal{B}(I)$ platí $c = c_{\square}$, potom $c = c_{\square\boxtimes}$.

Důkaz. Triviálně z definice operátorů $^{\square\boxtimes}, {}_{\square\boxtimes}$. □

Věta 3.7.
Pro koncept $c = \langle A, B\rangle \in \mathcal{B}(I)$ platí $A^{\uparrow_I} = A^{\uparrow_J}$, právě když $B^{\downarrow_I} = B^{\downarrow_J}$.

Důkaz. Pokud $B = A^{\uparrow_I} = A^{\uparrow_J}$, potom

$$A \subseteq A^{\uparrow_J\downarrow_J} = A^{\uparrow_I\downarrow_J} = B^{\downarrow_J} \subseteq B^{\downarrow_I} = A$$

a tedy $B^{\downarrow_I} = B^{\downarrow_J}$. Opačná implikace se dokazuje analogicky. □

V další části textu budeme potřebovat pojem *interval ve svazu*, který použijeme při zkoumání stability konceptů.

Definice 3.8 (interval ve svazu)**.**
Nechť $\langle L, \leq\rangle$ je svaz. Množina $K \subseteq L$ se nazývá interval ve svazu, nebo pro jednoduchost pouze interval, jestliže existují prvky $a, b \in L$ takové, že platí

$$K = \{k \in L \mid a \leq k \leq b\} \quad \textit{značeno} \quad [a, b].$$

Vlastnost stability má pro tvorbu cílového algoritmu velký význam a je zásadní přesně identifikovat stabilní koncepty. Pokud je koncept stabilní, víme, že se vyskytuje v obou konceptuálních svazech $\mathcal{B}(I)$, $\mathcal{B}(J)$. Identifikací stabilních konceptů zároveň identifikujeme i nestabilní koncepty, jelikož koncept je nestabilní, právě když není stabilní. Stabilní koncepty v $\mathcal{B}(I)$ lze popsat několika ekvivalentními způsoby, jak ukazuje následující teorém.

Teorém 3.9 (stabilní koncepty v $\mathcal{B}(I)$)**.**
Pro libovolný koncept $c \in \mathcal{B}(I)$ je následující ekvivalentní:

1. $c \notin [\gamma_I(x_0), \mu_I(y_0)]$,
2. *c je stabilní,*
3. $c = c^{\square}$,
4. $c = c_{\square}$,
5. $c^{\square} = c_{\square}$.

Důkaz. 1. $\implies$ 2.: Nechť $c = \langle A, B\rangle \notin [\gamma_I(x_0), \mu_I(y_0)]$. Potom buď platí $x_0 \notin A$ nebo $y_0 \notin B$. Pokud například, $x_0 \notin A$, potom dle věty 3.1, $B = A^{\uparrow_I} = A^{\uparrow_J}$, což je dle věty 3.7, právě když $B^{\downarrow_I} = B^{\downarrow_J}$. Tím pádem $c \in \mathcal{B}(J)$, takže c je stabilní. Důkaz pro případ $y_0 \notin B$ je duální.

2. $\implies$ 3. $\implies$ 4. $\implies$ 5.: Z definice, $c = \langle A, B\rangle$ je stabilní, právě když $A^{\uparrow_I} = A^{\uparrow_J}$ a $B^{\downarrow_I} = B^{\downarrow_J}$. Z věty 3.7 víme, že podmínka $A^{\uparrow_I} = A^{\uparrow_J}$ je ekvivalentní $B^{\downarrow_I} = B^{\downarrow_J}$. Použitím tohoto pozorování jsou dané implikace triviální.

5. $\implies$ 1.: Z toho, že $c^{\square} = c_{\square}$ máme $A^{\uparrow_J\downarrow_J} = B^{\downarrow_J}$ a $A^{\uparrow_J} = B^{\downarrow_J\uparrow_J}$, to ale znamená, že $c \in \mathcal{B}(J)$, což vylučuje právě koncepty z $[\gamma_I(x_0), \mu_I(y_0)]$, protože $\langle x_0, y_0\rangle \notin J$. □

Předchozí teorém udává popis stabilních konceptů v $\mathcal{B}(I)$ a to několika ekvivalentními způsoby. Obdobně lze popsat stabilní koncepty v $\mathcal{B}(J)$.

Teorém 3.10 (stabilní koncepty v $\mathcal{B}(J)$)**.**
Pro libovolný koncept $d \in \mathcal{B}(J)$ je následující ekvivalentní:

1. *d je stabilní,*
2. $d = d_{\boxtimes}$,
3. $d = d^{\boxtimes}$,
4. *$d^{\boxtimes}$ je stabilní,*
5. *$d_{\boxtimes}$ je stabilní.*

Důkaz. 1. $\Longrightarrow$ 2.: Z definice, $d = \langle C, D\rangle$ je stabilní, právě když $C^{\uparrow_J} = C^{\uparrow_I}$ a $D^{\downarrow_I} = D^{\downarrow_J}$. Dle definice operátoru $_{\boxtimes}$ rovnou dostáváme $d = d_{\boxtimes}$.

2. $\Longrightarrow$ 3.: Z toho, že $d = d_{\boxtimes}$ máme $D = C^{\uparrow_J} = C^{\uparrow_I}$ a $C = C^{\uparrow_I\downarrow_I}$, takže $C = D^{\downarrow_I}$. To ale znamená $d = d^{\boxtimes}$.

3. $\Longrightarrow$ 4.: Aby bylo $d^{\boxtimes}$ stabilní, musí platit $d^{\boxtimes} = \langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle \in \mathcal{B}(J)$. Z toho, že $d = d^{\boxtimes}$ máme $C = D^{\downarrow_J} = D^{\downarrow_I}$ a $D = D^{\downarrow_I\uparrow_I}$, takže $d^{\boxtimes} \in \mathcal{B}(J)$.

4. $\Longrightarrow$ 5.: Z definice, $d^{\boxtimes} = \langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle$ je stabilní, právě když $D^{\downarrow_I\uparrow_J} = D^{\downarrow_I\uparrow_I}$ a $D^{\downarrow_I\uparrow_I\downarrow_J} = D^{\downarrow_I\uparrow_I\downarrow_I} = D^{\downarrow_I}$, takže $D^{\downarrow_I} = D^{\downarrow_J} = C$. Dále z toho, že $d^{\boxtimes}$ je stabilní, musí platit $d^{\boxtimes} = \langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle \in \mathcal{B}(J)$. Dohromady dostáváme $d_{\boxtimes} = \langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle \in \mathcal{B}(J)$.

5. $\Longrightarrow$ 1.: Z definice, $d_{\boxtimes} = \langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle$ je stabilní, právě když $C^{\uparrow_I\downarrow_I\uparrow_J} = C^{\uparrow_I\downarrow_I\uparrow_I} = C^{\uparrow_I}$ a $C^{\uparrow_I\downarrow_J} = C^{\uparrow_I\downarrow_I}$, takže v důsledku věty 3.1 $C^{\uparrow_I} = C^{\uparrow_J} = D$. Dále z toho, že $d_{\boxtimes}$ je stabilní musí platit $d_{\boxtimes} = \langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle \in \mathcal{B}(J)$. Dohromady dostáváme $\langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle = \langle C^{\uparrow_J\downarrow_I}, C^{\uparrow_J}\rangle = \langle D^{\downarrow_I}, D\rangle \in \mathcal{B}(J)$, takže také $D^{\downarrow_I} = D^{\downarrow_J}$ a to dle definice znamená, že d je stabilní. □

Z předchozích poznatků plyne důležité pozorování o struktuře nestabilních konceptů v $\mathcal{B}(I)$.

Důsledek 3.11.
Nestabilní koncepty v $\mathcal{B}(I)$ tvoří právě interval $[\gamma_I(x_0), \mu_I(y_0)]$.

Předchozí také znamená, že koncepty z $\mathcal{B}(I) \setminus \mathcal{B}(J)$ jsou právě koncepty z $[\gamma_I(x_0), \mu_I(y_0)]$, což je vítaná vlastnost. Pokud například chceme prozkoumat všechny nestabilní koncepty, tak vlastně zkoumáme jeden daný interval. Je nutné si uvědomit, že pro nestabilní koncepty v $\mathcal{B}(J)$ podobné tvrzení neplatí a tedy netvoří interval. To je také jeden z důvodů, proč je výhodnější vycházet ze zúplnění $C(I)$ a ne z $C(J)$. Dalším důvodem je snadnější popis možných změn, kterými může koncept projít po odebrání dané incidence.

Dále se zaměříme na nestabilní koncepty v $\mathcal{B}(J)$. Následující teorém ukazuje, že pro každý takový koncept existuje právě jeden nestabilní koncept v $\mathcal{B}(I)$, který se na něj zobrazí pomocí operátoru $^{\square}$ nebo $_{\square}$.

Teorém 3.12.
Pro libovolný nestabilní koncept $d = \langle C, D\rangle \in \mathcal{B}(J)$ existuje právě jeden $c \in [\gamma_I(x_0), \mu_I(y_0)]$ takový, že

$$c = d^{\boxtimes} = d_{\boxtimes} \quad \textit{a platí} \quad c^{\square} = d \textit{ nebo } c_{\square} = d.$$

Důkaz. Z toho, že d je nestabilní, musí platit buď $C^{\uparrow_I} \neq C^{\uparrow_J}$ nebo $D^{\downarrow_I} \neq D^{\downarrow_J}$.

Předpokládejme, že $C^{\uparrow_I} \neq C^{\uparrow_J}$. Z věty 3.1, $x_0 \in C$, $y_0 \notin D$ a $C^{\uparrow_I} = D \cup \{y_0\}$. Takže, $d^{\boxtimes} = d_{\boxtimes} = \langle C, D \cup \{y_0\}\rangle$. Položme $c = d^{\boxtimes}$ a ihned dle definice dostáváme $c^{\square} = d$.

Podobně pro případ $D^{\downarrow_I} \neq D^{\downarrow_J}$. Z věty 3.1, $x_0 \notin C$, $y_0 \in D$ a $D^{\downarrow_I} = C \cup \{x_0\}$. Takže, $d^{\boxtimes} = d_{\boxtimes} = \langle C \cup \{x_0\}, D\rangle$. Položme $c = d_{\boxtimes}$ a ihned dle definice dostáváme $c_{\square} = d$. Jednoznačnost je triviální. □

Dle předchozího teorému, pro každý nestabilní koncept v $\mathcal{B}(J)$ existuje právě jeden nestabilní koncept v $\mathcal{B}(I)$ takový, že mezi sebou mají vztah daný operátory $^{\square}, ^{\boxtimes}$ nebo $_{\square}, _{\boxtimes}$. Navíc v důkazu tohoto teorému je specifikován i přesný popis těchto nestabilních konceptů v $\mathcal{B}(I)$.

Dále je nutné prozkoumat opačnou stránku věci, zda pro každý nestabilní koncept v $\mathcal{B}(I)$, splňující danou podmínku, existuje odpovídající nestabilní koncept v $\mathcal{B}(J)$.

Věta 3.13.
Nechť $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ *takový, že* $c = c_{\square\boxtimes}$, *potom*

$$d = c_{\square} = \langle A \setminus \{x_0\}, B\rangle \in \mathcal{B}(J) \quad \textit{a také} \quad c = d_{\boxtimes}.$$

Nechť $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ *takový, že* $c = c^{\square\boxtimes}$, *potom*

$$d = c^{\square} = \langle A, B \setminus \{y_0\}\rangle \in \mathcal{B}(J) \quad \textit{a také} \quad c = d^{\boxtimes}.$$

Důkaz. Dokážeme první část, důkaz druhé je duální. Dle věty 3.1, dostáváme $B^{\downarrow_J} = B^{\downarrow_I} \setminus \{x_0\} = A \setminus \{x_0\}$. Z toho, že $c = c_{\square\boxtimes}$ máme $B = B^{\downarrow_J\uparrow_I}$. Takže $B = B^{\downarrow_J\uparrow_I} = (A \setminus \{x_0\})^{\uparrow_I} = (A \setminus \{x_0\})^{\uparrow_J}$, tím pádem $d \in \mathcal{B}(J)$ a také $c = d_{\boxtimes}$. □

Důsledkem předchozích tvrzení je skutečnost, že každý nestabilní koncept v $\mathcal{B}(J)$ lze vyjádřit pomocí operátorů $^{\square}, _{\square}, ^{\boxtimes}, _{\boxtimes}$ z právě jednoho konceptu v $[\gamma_I(x_0), \mu_I(y_0)]$.

Důsledek 3.14.
Platí $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$, $c = c_{\square\boxtimes}$, *p.k.* $c_{\square} = \langle A \setminus \{x_0\}, B\rangle$, $c = d_{\boxtimes}$.
Platí $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$, $c = c^{\square\boxtimes}$, *p.k.* $c^{\square} = \langle A, B \setminus \{y_0\}\rangle$, $c = d^{\boxtimes}$.
Nechť $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ *takový, že* $c \neq c^{\square\boxtimes}$, $c \neq c_{\square\boxtimes}$, *potom platí*

$$\nexists \langle C, D\rangle \in \mathcal{B}(J) : \; C \subseteq A, \; D \subseteq B.$$

Příklad 3.15.
Příklady kontextů s různými typy konceptů vzhledem k operátorům $^{\square\boxtimes}, _{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	?	×	×
x_1			
x_2			

(a) Nejmenší koncept je pevným bodem obou operátorů.

	y_1	y_2	y_3	y_0
x_0	×	×	×	?
x_2			×	×
x_3		×		×
x_4	×			×

(b) Dva koncepty jsou pevnými body obou operátorů.

	y_1	y_2	y_0
x_0		×	?
x_1		×	×
x_2		×	

(c) Koncept je pevným bodem $_{\square\boxtimes}$, ale ne $^{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	?	×	
x_1	×	×	×
x_2			

(d) Koncept je pevným bodem $^{\square\boxtimes}$, ale ne $_{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	?	×	
x_1	×		
x_2			

(e) Kombinace předchozích dvou příkladů.

	y_0	y_2	y_3
x_0	?		
x_2	×	×	
x_3			

(f) Koncept není pevným bodem ani jednoho operátoru.

	y_1	y_2	y_3	y_4	y_0
x_0		×		×	?
x_1			×	×	×
x_2				×	
x_3	×	×			×
x_4		×			

(g) Dva koncepty nejsou pevným bodem ani jednoho operátoru.

Je dobré si všimnout, že formální kontexty z příkladů 1b, 1g lze dále rozšiřovat zjevným způsobem a tím získat příklady kontextů, které budou obsahovat libovolný počet konceptů s příslušnými vlastnostmi.

3.1.2. Algoritmy pro odvození počtu konceptů možných zúplnění

Nyní již máme dostatečně prozkoumané základní pojmy k sestavení algoritmu pro odvození počtu konceptů $\mathcal{B}(J)$ na základě $\mathcal{B}(I)$. Je třeba projít všechny nestabilní koncepty v $\mathcal{B}(I)$ a určit, zda jsou pevné body operátorů $^{\square\boxtimes}, _{\square\boxtimes}$. Důležité jsou následující dvě situace:

1. Nestabilní koncept $c \in \mathcal{B}(I)$ je pevným bodem obou operátorů $^{\square\boxtimes}, _{\square\boxtimes}$. Takže existují dva koncepty v $\mathcal{B}(J)$, které mají vztah k c daný operátory $^{\boxtimes}, _{\boxtimes}$.

2. Nestabilní koncept $c \in \mathcal{B}(I)$ není pevným bodem ani jednoho z operátorů $^{\square\boxtimes}, _{\square\boxtimes}$. Takže neexistuje žádný koncept v $\mathcal{B}(J)$, který by měl vztah k c daný operátory $^{\boxtimes}, _{\boxtimes}$.

Algoritmus 3 Počet konceptů $\mathcal{B}(J)$ na základě $\mathcal{B}(I)$.

```
procedure DeriveConceptCount(B(I))
    count ← |B(I)|;
    for all c ∈ [γ_I(x_0), μ_I(y_0)] do          ▷ Pro všechny nestabilní koncepty.
        if c = c^{□⊠} and c = c_{□⊠} then          ▷ Koncept se rozdělí.
            count ← count + 1;
        else if c ≠ c^{□⊠} and c ≠ c_{□⊠} then     ▷ Koncept zmizí.
            count ← count − 1;
        end if
    end for
    return count;
end procedure
```

Dále můžeme představit algoritmus pro výpočet zúplnění neúplného formálního kontextu na základě počítání konceptů. První verze počítá se znalostí $\mathcal{B}(I)$. Druhá verze naopak tuto znalost nepředpokládá a počítá jen nutnou část $\mathcal{B}(I)$ k odvození rozdílu počtu konceptů mezi $\mathcal{B}(I)$ a $\mathcal{B}(J)$.

Algoritmus 4 Zúplnění na základě počítání konceptů.

```
procedure GetMissingValueV1(B(I))
    BJCount ← DeriveConceptCount(B(I));        ▷ Použití algoritmu 3.
    if |B(I)| < BJCount then
        return '×';
    else if |B(I)| > BJCount then
        return '−';
    end if
    return '?';
end procedure

procedure GetMissingValueV2(C)
    BJdif ← 0;
    for all c ∈ [γ_I(x_0), μ_I(y_0)] do        ▷ Např. pomocí UpperNeighbor.
        if c = c^{□⊠} and c = c_{□⊠} then          ▷ Koncept se rozdělí.
            BJdif ← BJdif + 1;
        else if c ≠ c^{□⊠} and c ≠ c_{□⊠} then     ▷ Koncept zmizí.
            BJdif ← BJdif − 1;
        end if
    end for
    if BJdif > 0 then
        return '×';
    else if BJdif < 0 then
        return '−';
    end if
    return '?';
end procedure
```

3.1.3. Experimenty

Pomocí algoritmu 4 jsem provedl několik experimentů. První experimenty se týkaly náhodných dat s různou hustotou. Zvolil jsem tři omezení na hustotu generovaných dat. Pro každé zvolené omezení hustoty jsem vygeneroval dva tisíce náhodných formálních kontextů $C_i = \langle X_i, Y_i, I_i \rangle$, kde $|X_i| \in [5, 15]$, $|Y_i| \in [5, 10]$. Pro každý vygenerovaný formální kontext jsem postupně umístil neznámou hodnotu na všechny možné pozice a aplikoval algoritmus 4. Poté jsem daný formální kontext redukoval (1.2.) a provedl stejný postup jako v předchozím případě. Redukci jsem prováděl za účelem zjistit, zda je tato metoda závislá na redundanci dat. Celkově bylo provedeno okolo pět set tisíc testů nad náhodnými daty.

Hustota	Úspěšnost	Nerozhodnuto
0,13	51 %	40 %
0,50	41 %	27 %
0,87	46 %	31 %

(a) Výsledky experimentů nad náhodnými daty.

Hustota	Úspěšnost	Nerozhodnuto
0,13	24 %	45 %
0,50	32 %	23 %
0,87	9 %	13 %

(b) Výsledky experimentů nad klarifikovanými a redukovanými daty z experimentů (a).

Tabulka 3: Výsledky experimentů zúplňování pomocí počítání konceptů nad náhodnými daty různé hustoty.

Další experimenty, které jsem provedl, se týkaly známých ukázkových kontextů a vybraných reálných dat. Použité formální kontexty lze nalézt na přiloženém CD nebo ve zdrojích uvedených u jednotlivých formálních kontextů.

1. Datasety převzaté z *UCI machine learning repository* [10]:
   (a) *Zoo* - zvířata v zoo a jejich vlastnosti;
   (b) *Shuttle* - druhy raketoplánů a vhodnost různých druhů autopilota;
   (c) *Lenses* - kontaktní čočky a jejich vlastnosti;
   (d) *Post–operative patients* - stav pacientů po operaci.
2. Formální kontexty převzaté z článku Willeho [14]:
   (a) *Lattices* - svazy a jejich vlastnosti;
   (b) *Tea ladies*.
3. Formální kontexty převzaté z článku Bělohlávka a Trnečky [15]:
   (a) *Sports* - sporty a jejich vlastnosti;
   (b) *Drinks* - nápoje a jejich vlastnosti;

4. *Lives in water* - převzato z knihy Gantera a Willeho [3].

5. *Digits* - převzato z článku Willeho a Stahla [16], čísla a jejich vybrané vlastnosti.

Postup při provádění těchto experimentů, byl stejný jako v případě experimentů nad náhodnými daty. Neznámá hodnota byla umístěna na všechny možné pozice a poté byla vypočítána hodnota pro zúplnění pomocí algoritmu 4.

Dataset	Objektů	Atributů	**Úspěšnost**	Nerozhodnuto
Zoo	101	28	**84 %**	10 %
Shuttle	15	23	**75 %**	10 %
Lenses	24	12	**41 %**	17 %
Post–operative patients	90	25	**60 %**	15 %
Lattices	14	16	**86 %**	6 %
Tea ladies	18	14	**62 %**	13 %
Sport	28	16	**81 %**	11 %
Drinks	68	25	**87 %**	6 %
Lives in water	8	9	**56 %**	7 %
Digits	10	7	**23 %**	21 %

Tabulka 4: Výsledky experimentů zúplňování pomocí počítání konceptů nad ukázkovými a reálnými daty.

Výsledky experimentů napovídají, že nad náhodnými daty dává tato metoda náhodné výsledky, což je očekávané. Kdyby tomu tak nebylo, znamenalo by to, že buď jsou experimenty špatně navrženy nebo je na této metodě něco podezřelého. To proto, že z náhodných dat by obecně nemělo být možné vyčíst jakoukoli strukturu (vzor v datech) a tím pádem by tato metoda měla dávat nahodilé výsledky.

Experimenty nad ukázkovými a vybranými reálnými daty vykazují poměrně zajímavou úspěšnost této metody. Situace, kdy nebylo možné rozhodnout, které zúplnění zvolit, nastala v průměru u 12 % experimentů. Úspěšnost zúplnění byla v průměru okolo 65 % a ve zbylých 23 % se zvolená hodnota lišila od původní. Po klarifikaci a redukci daných kontextů se úspěšnost snížila o 1-15 %, přičemž průměrná úspěšnost se snížila o 7 %. Procento situací, kdy nebylo možné rozhodnout, se změnilo jen nepatrně (±3%). V průměru se úspěšnost této metody po redukci zhoršila, což naznačuje, že tato metoda je ovlivněna redundancí ve vstupních datech.

3.2. Transformace konceptuálního svazu

Při zkoumání metody zúplňování neúplného formálního kontextu pomocí počítání konceptů se postupně ukázalo, že tato metoda vede ke zkoumání změny konceptuálního svazu při změně dané incidence. Na začátku mého zkoumání jsem se na tento problém snažil dívat jako na přidání incidence. Podrobnějším zkoumáním jsem došel k závěru, že pohled na tento problém jako na odebrání incidence je výhodnější. Cílem bylo vytvořit algoritmus, který pomocí inkrementálních úprav transformuje $\mathcal{B}(I)$ na $\mathcal{B}(J)$. V celé této části tedy předpokládáme znalost konceptuálního svazu $\mathcal{B}(I)$, který chceme transformovat na $\mathcal{B}(J)$.

3.2.1. Transformace konceptů

Z předchozí části již máme zavedené vhodné pojmy, které využijeme i při řešení tohoto problému. Jednoduchý algoritmus na transformaci pouze konceptů (bez informace o sousedech) $\mathcal{B}(I)$ na koncepty $\mathcal{B}(J)$ je v podstatě shrnutí a ucelení výsledků z předchozí kapitoly. Obtížnější částí je transformace informace o sousedech konceptů, ke které je potřeba dokázat několik dalších tvrzení.

Na $\mathcal{B}(J)$ můžeme nahlížet jako na $\mathcal{B}(I)$ po určitých transformacích. Z předchozího dostáváme, že všechny možné transformace konceptů, které mohou nastat při odebrání dané incidence, lze popsat pomocí operátorů $^{\square\boxtimes}, _{\square\boxtimes}$. Konkrétně pro koncept $c \in [\gamma_I(x_0), \mu_I(y_0)]$ se jedná o následující čtyři možné transformace:

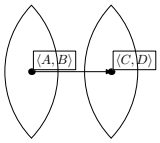
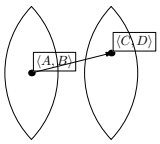
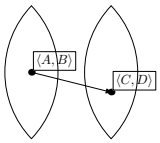
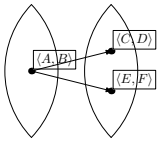
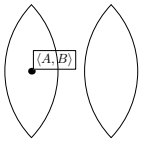
1. extent c se zmenší o x_0,
2. intent c se zmenší o y_0,
3. c se rozdělí na dva nové koncepty,
4. c zmizí a žádná jeho část nebude konceptem v $\mathcal{B}(J)$.

Tabulka 5 shrnuje všechny představené výsledky týkající se vztahů mezi koncepty $\mathcal{B}(I)$, $\mathcal{B}(J)$.

Předchozí poznatky vedou na jednoduchou metodu konstrukce $\mathcal{B}(J)$ z $\mathcal{B}(I)$. Pro každý $c \in \mathcal{B}(I)$ je potřeba udělat následující:

1. Pokud je c stabilní, přidej c do $\mathcal{B}(J)$.
2. Pokud c není stabilní, pak každý nestabilní prvek z množiny $\{c^{\square}, c_{\square}\}$ přidej do $\mathcal{B}(J)$.

Touto metodou je zajištěno, že všechny koncepty patřící do $\mathcal{B}(J)$ budou při konstrukci přidány a každý z nich bude přidán právě jednou.

	Obrázek	Kontext	$c = \langle A, B\rangle \in \mathcal{B}(I)$	$d = \langle C, D\rangle \in \mathcal{B}(J)$ $(e = \langle E, F\rangle \in \mathcal{B}(J))$
$c, d \in \mathcal{B}(I) \cap \mathcal{B}(J)$		$\begin{array}{c\|ccc} & y_1 & y_2 & y_0 \\ \hline x_0 & & & ? \\ x_1 & \times & \times & \\ x_2 & \times & \times & \end{array}$	c je stabilní, $c \notin [\gamma_I(x_0), \mu_I(y_0)]$	d je stabilní, $d = d^{\boxtimes} = d_{\boxtimes}$
$c \in \mathcal{B}(I) \setminus \mathcal{B}(J)$, $d \in \mathcal{B}(J) \setminus \mathcal{B}(I)$		$\begin{array}{c\|ccc} & y_1 & y_2 & y_0 \\ \hline x_0 & & \times & ? \\ x_1 & \times & \times & \times \\ x_2 & & & \end{array}$	zmenší se intent, $c = c^{\square\boxtimes},\ c \neq c_{\square\boxtimes}$	zvětší se intent, $y_0 \notin D,\ d \neq d_{\boxtimes} = c$
$c \in \mathcal{B}(I) \setminus \mathcal{B}(J)$, $d \in \mathcal{B}(J) \setminus \mathcal{B}(I)$		$\begin{array}{c\|ccc} & y_1 & y_2 & y_0 \\ \hline x_0 & & \times & ? \\ x_1 & & \times & \times \\ x_2 & & \times & \end{array}$	zmenší se extent, $c \neq c^{\square\boxtimes},\ c = c_{\square\boxtimes}$	zvětší se extent, $x_0 \notin C,\ d \neq d^{\boxtimes} = c$
$c \in \mathcal{B}(I) \setminus \mathcal{B}(J)$, $d \in \mathcal{B}(J) \setminus \mathcal{B}(I)$		$\begin{array}{c\|ccc} & y_1 & y_2 & y_0 \\ \hline x_0 & & \times & ? \\ x_1 & & \times & \times \\ x_2 & & & \end{array}$ $\begin{array}{c\|ccc} & y_1 & y_2 & y_0 \\ \hline x_0 & \times & \times & ? \\ x_1 & & & \\ x_2 & & & \end{array}$	c se rozdělí, $c = c^{\square\boxtimes},\ c = c_{\square\boxtimes}$	d a e se spojí, $d_{\boxtimes} = e^{\boxtimes} = c$
$c \in \mathcal{B}(I) \setminus \mathcal{B}(J)$		$\begin{array}{c\|ccc} & y_1 & y & y_3 \\ \hline x_1 & & & \\ x & & ? & \\ x_3 & & & \end{array}$ $\begin{array}{c\|ccc} & y_1 & y_2 & y_0 \\ \hline x_0 & & \times & ? \\ x_1 & \times & \times & \times \\ x_2 & & \times & \end{array}$	c zmizí, $c \neq c^{\square\boxtimes},\ c \neq c_{\square\boxtimes}$	–

Tabulka 5: Shrnutí vztahů mezi koncepty $\mathcal{B}(I)$ a $\mathcal{B}(J)$.

Algoritmus 5 Transformace $\mathcal{B}(I)$, bez informace o struktuře, na $\mathcal{B}(J)$.

```
procedure TransformConcepts(B(I))
    for all c = ⟨A, B⟩ ∈ [γ_I(x_0), μ_I(y_0)] do
        B(I) ← B(I) \ {c};
        if c = c_{□⊠} then
            B(I) ← B(I) ∪ {c_□};
        end if
        if c = c^{□⊠} then
            B(I) ← B(I) ∪ {c^□};
        end if
    end for
end procedure
```

3.2.2. Transformace struktury konceptuálního svazu

Nyní se zaměříme na rozšíření algoritmu na transformaci konceptů $\mathcal{B}(I)$ na koncepty $\mathcal{B}(J)$ tak, aby pracoval i s informací o struktuře (sousedských vztazích). Cílem je tedy vytvořit algoritmus, který konceptuální svaz $\mathcal{B}(I)$ transformuje na $\mathcal{B}(J)$ a to včetně struktury. Využijeme již zavedené operátory ${}^{\square}{}_{\square}, {}^{\boxtimes}{}_{,\boxtimes}$ a podíváme se na jejich strukturální vlastnosti. Zejména důležité jsou uzávěrové operátory ${}^{\square\boxtimes}{}_{,\square\boxtimes}$. Následující věta se zaměřuje právě na tyto operátory. Konkrétně, pokud se výsledek aplikace operátoru ${}^{\square\boxtimes}$ (${}_{\square\boxtimes}$) na koncept c liší od c, jaké jsou vlastnosti $c^{\square\boxtimes}$ ($c_{\square\boxtimes}$).

Věta 3.16.
Nechť $c \in \mathcal{B}(I),\ c \neq c^{\square\boxtimes}$, potom $c^{\square\boxtimes}$ je horní soused konceptu c.
Nechť $c \in \mathcal{B}(I),\ c \neq c_{\square\boxtimes}$, potom $c_{\square\boxtimes}$ je dolní soused konceptu c.

Důkaz. Triviálně z definice operátorů ${}^{\square\boxtimes}{}_{,\square\boxtimes}$. □

Aplikací operátorů ${}^{\square\boxtimes}{}_{,\square\boxtimes}$ na koncept c tedy vždy získáme jedno z následujících:

- $c = c^{\square\boxtimes}$ – c je pevný bod operátoru ${}^{\square\boxtimes}$,
- $c = c_{\square\boxtimes}$ – c je pevný bod operátoru ${}_{\square\boxtimes}$,
- $c^{\square\boxtimes}$ je horní soused c,
- $c_{\square\boxtimes}$ je dolní soused c.

Tím pádem víme, že $[c_{\square\boxtimes}, c^{\square\boxtimes}]$ tvoří neprázdný interval. Je tedy vhodné zamyslet se nad vlastnostmi konceptů v tomto intervalu.

Věta 3.17.
Každý koncept z $[c_{\square\boxtimes}, c^{\square\boxtimes}] \setminus \{c\}$ je stabilní.

Důkaz. Nechť $c = \langle A, B\rangle$. Předpokládejme, že $c' = \langle A', B'\rangle \in [c_{\square\boxtimes}, c^{\square\boxtimes}]$ není stabilní. Podle teorému 3.9, $x_0 \in A'$ a $y_0 \in B'$. Z toho, že $c' \geq c_{\square\boxtimes}$ a z faktu, že extent $c_{\square\boxtimes}$ je roven A nebo $A \setminus \{x_0\}$ dostáváme $A' \supseteq A$. Obdobně odvodíme $B' \supseteq B$ a tím pádem $c' = c$. □

Jelikož z předchozího již umíme identifikovat stabilní a nestabilní koncepty, tak v důsledku předchozích tvrzení dostáváme ihned některé vlastnosti konceptů z $[c_{\square\boxtimes}, c^{\square\boxtimes}] \setminus \{c\}$.

Důsledek 3.18.
Nechť $c \in \mathcal{B}(I)$ je koncept, pro který platí $c \neq c^{\square\boxtimes}$. Potom $c^{\square\boxtimes} \notin [\gamma(x_0), \mu(y_0)]$.
Nechť $c \in \mathcal{B}(I)$ je koncept, pro který platí $c \neq c_{\square\boxtimes}$. Potom $c_{\square\boxtimes} \notin [\gamma(x_0), \mu(y_0)]$.
Pro libovolný koncept $c \in \mathcal{B}(I)$ platí

$$\forall k \in [c_{\square\boxtimes}, c^{\square\boxtimes}] \setminus \{c\} \ : \ k \notin [\gamma(x_0), \mu(y_0)].$$

Jak je možné vidět v předchozích větách, často je třeba testovat, zda je koncept pevným bodem operátorů $^{\square\boxtimes}, _{\square\boxtimes}$. Pro řešení tohoto problému není nutné vypočítat samotnou aplikaci daného operátoru na příslušný koncept. Vystačíme si s jednoduššími testy, jak ukazují následující věty.

Věta 3.19.
Nechť $c = \langle A, B\rangle \in \mathcal{B}(I)$. Pokud $x_0 \notin A$, pak $c = c^{\square\boxtimes}$.
Nechť $c = \langle A, B\rangle \in \mathcal{B}(I)$. Pokud $y_0 \notin B$, pak $c = c_{\square\boxtimes}$.

Důkaz. Triviálně z věty 3.1 a definice operátorů $^{\square\boxtimes}, _{\square\boxtimes}$. □

Zvýšenou pozornost je nutné věnovat nestabilním konceptům, pro které je v následující větě uvedena vyčerpávající charakteristika pevných bodů operátorů $^{\square\boxtimes}, _{\square\boxtimes}$.

Věta 3.20.
Nechť $c = \langle A, B\rangle \in \mathcal{B}(I)$ je koncept, pro který platí $x_0 \in A$. Potom

$$c = c^{\square\boxtimes}, \ \text{právě když} \ (B \setminus \{y_0\})^{\downarrow_I} = A.$$

Nechť $c = \langle A, B\rangle \in \mathcal{B}(I)$ je koncept, pro který platí $y_0 \in B$. Potom

$$c = c_{\square\boxtimes}, \ \text{právě když} \ (A \setminus \{x_0\})^{\uparrow_I} = B.$$

Důkaz. Dokážeme první tvrzení, důkaz druhého je duální. Platí $c = c^{\square\boxtimes}$ p.k. $A = A^{\uparrow_J\downarrow_I}$. Dle věty 3.1, $A = A^{\uparrow_J\downarrow_I} = (A^{\uparrow_I} \setminus \{y_0\})^{\downarrow_I} = (B \setminus \{y_0\})^{\downarrow_I}$. □

Věta 3.21.
Pokud pro koncept $c = \langle A, B\rangle \in \mathcal{B}(I)$ platí $c \neq c^{\square\boxtimes}$, potom $y_0 \in B$.
Pokud pro koncept $c = \langle A, B\rangle \in \mathcal{B}(I)$ platí $c \neq c_{\square\boxtimes}$, potom $x_0 \in A$.

Důkaz. Dokážeme první tvrzení, důkaz druhého je duální. Předpokládejme, že $y_0 \notin B$. Obměnou věty 3.19 dostáváme $x_0 \in A$. Z předpokladu máme $B \setminus \{y_0\} = B$, takže dle věty 3.20 musí platit $B^{\downarrow_I} = (B \setminus \{y_0\})^{\downarrow_I} \supset A$, ale to je spor s tím, že c je koncept. □

Přímým důsledkem předchozích tvrzení je skutečnost, že koncepty, které nejsou pevné body operátoru ${}^{\square\boxtimes}$ nebo ${}_{\square\boxtimes}$ jsou nestabilní. Opačná implikace ovšem neplatí, tzn. nestabilní koncept může být pevným bodem obou těchto operátorů.

Důsledek 3.22.
Nechť $c \in \mathcal{B}(I)$ je koncept, pro který platí $c \neq c^{\square\boxtimes}$ nebo $c \neq c_{\square\boxtimes}$, potom

$$c \in [\gamma_I(x_0), \mu_I(y_0)].$$

Následující věta ukazuje další vlastnost operátorů ${}^{\square\boxtimes}, {}_{\square\boxtimes}$, kterou využijeme při tvorbě cílového algoritmu. Konkrétně udává, že pro koncept $c \in \mathcal{B}(I)$ všechny s ním srovnatelné stabilní koncepty jsou srovnatelné také s $c^{\square\boxtimes}$, $c_{\square\boxtimes}$.

Věta 3.23.
Nechť $c \in \mathcal{B}(I)$. Pro stabilní koncepty $c' \in \mathcal{B}(I)$ takové, že $c \leq c'$ platí $c^{\square\boxtimes} \leq c'$.
Nechť $c \in \mathcal{B}(I)$. Pro stabilní koncepty $c' \in \mathcal{B}(I)$ takové, že $c' \leq c$ platí $c' \leq c_{\square\boxtimes}$.

Důkaz. Dokážeme první tvrzení, důkaz druhého je duální. Nechť $c = \langle A, B\rangle$. Pokud $c = c^{\square\boxtimes}$, tvrzení platí triviálně. Předpokládejme, že $c \neq c^{\square\boxtimes}$. Podle poznatku 3.22, $c \in [\gamma_I(x_0), \mu_I(y_0)]$. Takže dle definice, $A^{\uparrow_J\downarrow_I\uparrow_I} = (A^{\uparrow_I} \setminus \{y_0\})^{\downarrow_I\uparrow_I} = (B \setminus \{y_0\})^{\downarrow_I\uparrow_I}$. Nechť $c' = \langle A', B'\rangle \in \mathcal{B}(I)$ je stabilní koncept, pro který platí $c \leq c'$ a tedy platí $B' \subseteq B$ a $y_0 \notin B'$, takže $B' \subseteq B \setminus \{y_0\}$. Tím pádem platí $B'^{\downarrow_I\uparrow_I} \subseteq (B \setminus \{y_0\})^{\downarrow_I\uparrow_I}$ a dostáváme $c^{\square\boxtimes} \leq c'$. □

Nyní se zaměřme na zkoumání struktury pevných bodů operátorů ${}^{\square\boxtimes}, {}_{\square\boxtimes}$. Známý výsledek v matematice se týká struktury pevných bodů uzávěrového (vnitřkového) operátoru nad danou množinou M v úplném svazu $\langle 2^M, \subseteq\rangle$. Tento výsledek udává, že daná struktura je úplný infimální podsvaz v případě uzávěrového operátoru a úplný supremální podsvaz v případě vnitřkového operátoru. Tuto skutečnost ověřuje pro náš konkrétní případ následující věta.

Věta 3.24.
Pro libovolné koncepty $c_i = \langle A_i, B_i\rangle \in \mathcal{B}(I),\ i \in I,\ c_i = {c_i}^{\square\boxtimes}$ platí

$$\bigwedge_{i\in I} c_i = c = c^{\square\boxtimes} = \langle A, B\rangle.$$

Pro libovolné koncepty $c_i = \langle A_i, B_i\rangle \in \mathcal{B}(I),\ i \in I,\ c_i = c_{i\square\boxtimes}$ platí

$$\bigvee_{i\in I} c_i = c = c_{\square\boxtimes} = \langle A, B\rangle.$$

Důkaz. Dokážeme první tvrzení, důkaz druhého je duální. Pokud $x_0 \notin A$, potom dle věty 3.19 máme $c = c^{\square\boxtimes}$ a tvrzení platí. Předpokládejme, že $x_0 \in A$. Dle základní věty FKA (věta 1.11), $B = (\cup_{i\in I} B_i)^{\downarrow_I\uparrow_I}$. Dle věty 3.20, pokud platí $((\cup_{i\in I} B_i)^{\downarrow_I\uparrow_I} \setminus \{y_0\})^{\downarrow_I} \neq A$, potom $c \neq c^{\square\boxtimes}$. Jediná možnost je $((\cup_{i\in I} B_i)^{\downarrow_I\uparrow_I} \setminus \{y_0\})^{\downarrow_I} \supset A$. Aby toto platilo, musí existovat objekt $x \in X$ takový, že $\{x\}^{\uparrow_I} \subseteq (\cup_{i\in I} B_i)^{\downarrow_I\uparrow_I} \setminus \{y_0\}$ a zároveň $x \notin A = \cap_{i\in I} A_i$, takže $\exists A_i : x \notin A_i$. To by ale znamenalo, že pro nějaké c_i platí $(B_i \setminus \{y_0\})^{\uparrow_I} \neq A_i$ a tedy $c_i \neq {c_i}^{\square\boxtimes}$, což je spor s předpokladem. □

Důsledek 3.25.
Pevné body uzávěrového operátoru $^{\square\boxtimes}$ *tvoří úplný infimální podsvaz* $\mathcal{B}(I)$.
Pevné body vnitřkového operátoru $_{\square\boxtimes}$ *tvoří úplný supremální podsvaz* $\mathcal{B}(I)$.

Tímto získáváme zajímavý vhled do struktury pevných bodů obou těchto operátorů. Přímým důsledkem je například skutečnost, že koncept, který není pevným bodem operátoru $^{\square\boxtimes}$ ($_{\square\boxtimes}$), má pouze jediného horního (dolního) souseda, který je pevným bodem daného operátoru a to právě $c^{\square\boxtimes}$ ($c_{\square\boxtimes}$).

Další užitečný poznatek o struktuře pevných bodů operátorů $c^{\square\boxtimes}$ ($c_{\square\boxtimes}$) je skutečnost, že pokud existuje nestabilní koncept, který je pevným bodem $c^{\square\boxtimes}$ ($c_{\square\boxtimes}$), tak všechny menší (větší) koncepty musí být také pevné body daného operátoru.

Věta 3.26.
Nechť $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ *je koncept, pro který platí* $c = c_{\square\boxtimes}$. *Potom*

$$\forall c_i \in \mathcal{B}(I),\ c \leq c_i,\ i \in I \quad \text{platí} \quad c_i = c_{i\square\boxtimes}.$$

Nechť $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ *je koncept, pro který platí* $c = c^{\square\boxtimes}$. *Potom*

$$\forall c_i \in \mathcal{B}(I),\ c_i \leq c,\ i \in I \quad \text{platí} \quad c_i = {c_i}^{\square\boxtimes}.$$

Důkaz. Dokážeme první tvrzení, důkaz druhého je duální. Předpokládejme, že tvrzení neplatí a za daného předpokladu platí $\exists c' = \langle A', B'\rangle : c \leq c',\ c' \neq c'_{\square\boxtimes}$. Dle věty 3.19 máme $y_0 \in B'$. Dále dle věty 3.20 musí platit $(A' \setminus \{x_0\})^{\uparrow_I} \supset B'$. Z toho, že $c = c_{\square\boxtimes}$ a z věty 3.20, $(A \setminus \{x_0\})^{\uparrow_I} = B$. Jinými slovy, $\exists y \in Y,\ y \notin B'$ takové, že $y \in (A' \setminus \{x_0\})^{\uparrow_I}$. Z toho, že $A \setminus \{x_0\} \subseteq A' \setminus \{x_0\}$ máme $B = (A \setminus \{x_0\})^{\uparrow_I} \supseteq (A' \setminus \{x_0\})^{\uparrow_I} \supseteq \{y\}$, takže $y \in B$ a dohromady s $x_0 \in A$ dostáváme $x_0 \in \{y\}^{\downarrow_I}$, což je spor s $y \notin B'$. □

Při transformaci $\mathcal{B}(I)$ na $\mathcal{B}(J)$ je nutné zkoumat, zda mezi určitými koncepty již neleží nějaký koncept. Je proto užitečné prozkoumat některé vlastnosti těchto konceptů. Následující věta ukazuje, že takovéto koncepty jsou vždy stabilní.

Věta 3.27.
Nechť $c \in [\gamma_I(x_0), \mu_I(y_0)]$. Potom každý koncept $c' \in \mathcal{B}(I)$, pro který platí

$$c' \leq c^{\square\boxtimes},\ c' \not\leq c$$

je stabilní.
Nechť $c \in [\gamma_I(x_0), \mu_I(y_0)]$. Potom každý koncept $c' \in \mathcal{B}(I)$, pro který platí

$$c_{\square\boxtimes} \leq c',\ c \not\leq c'$$

je stabilní.

Důkaz. Dokážeme první tvrzení, důkaz druhého je duální. Označme $c = \langle A, B\rangle,\ c' = \langle A', B'\rangle$. Z toho, že $c' \leq c^{\square\boxtimes}$, $A' \subseteq A^{\uparrow_J\downarrow_I} = (B \setminus \{y_0\})^{\downarrow_I}$. Dále z $c' \not\leq c$, $A' \not\subseteq A = B^{\downarrow_I}$. Celkově dostáváme $A' \subseteq (B \setminus \{y_0\})^{\downarrow_I}$ a $A' \not\subseteq B^{\downarrow_I}$. Takže A' musí obsahovat nějaké objekty, které nemají y_0 a tím pádem $y_0 \notin B'$ a v důsledku je c' stabilní. □

Důsledkem předchozích tvrzení jsou určitá omezení možných vlastností sousedů konceptů v $[\gamma_I(x_0), \mu_I(y_0)]$. Tímto způsobem jsme omezili možné situace, které je nutné při transformaci $\mathcal{B}(I)$ na $\mathcal{B}(J)$ řešit. Výsledné shrnutí je možné nalézt v tabulce 6. Hodnoty v jednotlivých polích tabulky 6 mají následující význam:

- $\nearrow$ – koncept v příslušném sloupci může být *horním* sousedem konceptu v příslušném řádku.
- $\swarrow$ – koncept v příslušném sloupci může být *dolním* sousedem konceptu v příslušném řádku.
- $\nearrow\swarrow$ – kombinace předchozích dvou případů.
- prázdné pole – příslušné koncepty nemohou být sousedé.

koncept / sousedé	$c' = c'^{\square\boxtimes}$, $c' = c'_{\square\boxtimes}$	$c' \neq c'^{\square\boxtimes}$, $c' = c'_{\square\boxtimes}$	$c' = c'^{\square\boxtimes}$, $c' \neq c'_{\square\boxtimes}$	$c' \neq c'^{\square\boxtimes}$, $c' \neq c'_{\square\boxtimes}$
$c = c^{\square\boxtimes}, c = c_{\square\boxtimes}$	$\nearrow\swarrow$	$\nearrow$	$\swarrow$	
$c \neq c^{\square\boxtimes}, c = c_{\square\boxtimes}$	$\swarrow$	$\nearrow\swarrow$	$\swarrow$	$\swarrow$
$c = c^{\square\boxtimes}, c \neq c_{\square\boxtimes}$	$\nearrow$	$\nearrow$	$\nearrow\swarrow$	$\nearrow$
$c \neq c^{\square\boxtimes}, c \neq c_{\square\boxtimes}$		$\nearrow$	$\swarrow$	$\nearrow\swarrow$

Tabulka 6: Všechny možné sousedské vztahy nestabilních konceptů v $\mathcal{B}(I)$.

3.2.3. Výsledný algoritmus

Nyní již máme dokázáno vše potřebné pro představení algoritmu na transformaci $\mathcal{B}(I)$ na $\mathcal{B}(J)$. Začneme slovním popisem kritické části algoritmu, kterou je transformace struktury konceptuálního svazu. V tomto popisu jsou záměrně vynechány souměrné části jednotlivých případů.

Označme $c = \langle A, B\rangle$ libovolný nestabilní koncept z $\mathcal{B}(I)$ ($c \in [\gamma_I(x_0), \mu_I(y_0)]$).

- Pokud $c = c^{\square\boxtimes}$, $c = c_{\square\boxtimes}$, potom se c rozdělí na dva koncepty $d_1 \leq d_2$.
  - Koncept d_1 bude spodním sousedem konceptu d_2.
  - Extent c se stane extentem d_2. Intent c se stane intentem d_1.
  - Takže $d_1 = \langle A_{d_1}, B_{d_1}\rangle = \langle A \setminus \{x_0\}, B\rangle = c_\square$ a dále $d_2 = \langle A_{d_2}, B_{d_2}\rangle = \langle A, B \setminus \{y_0\}\rangle = c^\square$.
  - Pokud pro nějakého spodního souseda c_l platí $c_l = {c_l}^{\square\boxtimes}$, $c_l \neq c_{l\square\boxtimes}$ (c_l ztratí y_0 z intentu), potom bude spodním sousedem d_2. Dále je nutné ověřit, zda d_1 a $c_{l\square\boxtimes}$ budou sousedé. Určitě platí $c_{l\square\boxtimes} \leq d_1$, ale může existovat stabilní koncept k takový, že $c_{l\square\boxtimes} \leq k \leq d_1$.
  - Pokud pro nějakého horního souseda c_u platí $c_u \neq {c_u}^{\square\boxtimes}$, $c_u = c_{u\square\boxtimes}$ (c_u ztratí x_0 z extentu), potom bude horním sousedem d_1. Opět je nutné ověřit, zda d_2 a ${c_u}^{\square\boxtimes}$ budou sousedé. Platí $d_2 \leq {c_u}^{\square\boxtimes}$, ale může existovat stabilní koncept k takový, že $d_2 \leq k \leq {c_u}^{\square\boxtimes}$.
  - Pokud pro nějakého nestabilního souseda c_n platí $c_n = {c_n}^{\square\boxtimes}$, $c_n = c_{n\square\boxtimes}$, tedy platí pro něj to samé jako pro c, c_n se rozdělí na d_{n_1}, d_{n_2}. V takovém případě budou d_1, d_{n_1} a d_2, d_{n_2} sousedé.
  - Pro všechny ostatní horní sousedy $c_i = \langle A_i, B_i\rangle$ konceptu c platí $A \subseteq A_i$. Jelikož $A_{d_2} = A$, tak $A_{d_1} \subseteq A_{d_2} \subseteq A_i$. Takže všechny tyto koncepty budou horní sousedé konceptu d_2.
  - Pro všechny ostatní dolní sousedy c_i konceptu c platí $B \subseteq B_i$. Z toho, že $B_{d_1} = B$, dostáváme $B_{d_2} \subseteq B_{d_1} \subseteq B_i$. Takže všechny tyto koncepty budou dolní sousedé konceptu d_1.
- Pokud $c \neq c^{\square\boxtimes}$ a $c = c_{\square\boxtimes}$, potom c ztratí x_0 ze svého extentu.
  - Označme transformovaný c jako $d = \langle C, D\rangle = c_\square = \langle A \setminus \{x_0\}, B\rangle$.
  - Pokud pro nějakého spodního souseda c_l platí $c_l = {c_l}^{\square\boxtimes}$, $c_l \neq c_{l\square\boxtimes}$ (c_l ztratí y_0 z intentu), potom se c_l a d stanou nesrovnatelnými. Musíme ale ověřit, zda $c^{\square\boxtimes}$ a $c_{l\square\boxtimes}$ mají být sousedé. Platí $c_{l\square\boxtimes} \leq c^{\square\boxtimes}$, ale může existovat stabilní koncept k takový, že $c_{l\square\boxtimes} \leq k \leq c^{\square\boxtimes}$.

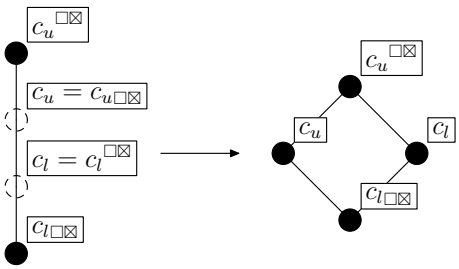
  - Pro všechny ostatní horní (dolní) sousedy c_i konceptu c platí $B_i \subseteq B$ ($B \subseteq B_i$). Jelikož $D = B$, tak platí $B_i \subseteq D$ ($D \subseteq B_i$).

- Pokud $c = c^{\square\boxtimes}$ a $c \neq c_{\square\boxtimes}$, potom c ztratí y_0 ze svého intentu.
  - Označme transformovaný c jako $d = \langle C, D\rangle = d^{\square} = \langle A, B \setminus \{y_0\}\rangle$.
  - Pro všechny stabilní horní (dolní) sousedy c_i of c platí $A \subseteq A_i$ ($A_i \subseteq A$). Z toho, že $C = A$, dostáváme $C \subseteq A_i$ ($A_i \subseteq C$).

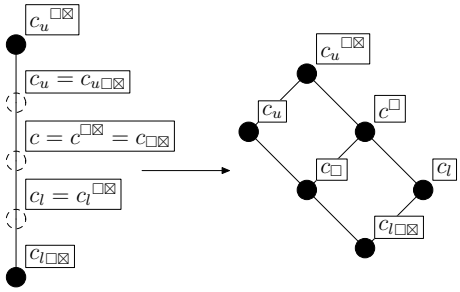
- Pokud $c \neq c^{\square\boxtimes}$ a $c \neq c_{\square\boxtimes}$, potom c zmizí a žádná jeho část nebude konceptem v $\mathcal{B}(J)$.
  - Musíme ověřit, zda $c_{\square\boxtimes}$ a $c^{\square\boxtimes}$ budou sousedé. Koncepty $c_{\square\boxtimes}$ a $c^{\square\boxtimes}$ jsou srovnatelné, ale může mezi nimi již být nějaký stabilní koncept.
  - Označme U množinu všech horních sousedů konceptu c, vyjma $c^{\square\boxtimes}$. Z důsledku 3.25 víme, že žádný koncept z U nemůže být pevným bodem operátoru $^{\square\boxtimes}$, jinak by c musel být pevným bodem $^{\square\boxtimes}$, protože pevné body operátoru $^{\square\boxtimes}$ tvoří úplný infimální podsvaz $\mathcal{B}(I)$.
  - Označme L množinu všech dolních sousedů konceptu c, vyjma $c_{\square\boxtimes}$. Podobně jako v předchozím bodu dostáváme, že žádný koncept z L nemůže být pevným bodem operátoru $_{\square\boxtimes}$.
  - Žádné dva koncepty $c_u \in U,\ c_l \in L$ nebudou sousedé.
    Koncepty z U nejsou pevné body operátoru $^{\square\boxtimes}$, takže buď zmizí (nejsou ani pevné body operátoru $_{\square\boxtimes}$) nebo ztratí x_0 z extentu.
    Obdobně, koncepty z L nejsou pevné body operátoru $_{\square\boxtimes}$, takže buď zmizí (nejsou pevné body operátoru $^{\square\boxtimes}$) nebo ztratí y_0 z intentu.
    Dále víme, že extenty konceptů z L jsou podmnožiny extentů konceptů z U, které obsahují x_0, protože nejsou pevné body operátoru $_{\square\boxtimes}$. Pokud tedy odebereme x_0 z extentů konceptů z U, tak koncepty z $U,\ L$ se stanou nesrovnatelnými.

    Dokonce ani není nutné ověřovat, zda některé koncepty z U (L) a někteří stabilní spodní (horní) sousedé konceptů L (U) budou sousedé, protože důsledkem věty 3.23 mezi nimi určitě leží koncept $c_{\square\boxtimes}$ ($c^{\square\boxtimes}$). Předchozí není nutné ověřovat ani pro nestabilní sousedy daných konceptů, protože ti mohou být dle tabulky 6 pouze jednoho typu, pro který již víme, že takovéto koncepty nebudou sousedé.
  - Platí $\forall c_l \in L :\ c_l \leq c \leq c^{\square\boxtimes}$, ale musíme ověřit, zda mezi nimi již neleží nějaký koncept.
  - Obdobně, platí $\forall c_h \in U : c_{\square\boxtimes} \leq c \leq c_h$, ale opět musíme ověřit, jestli mezi nimi již neleží nějaký koncept.

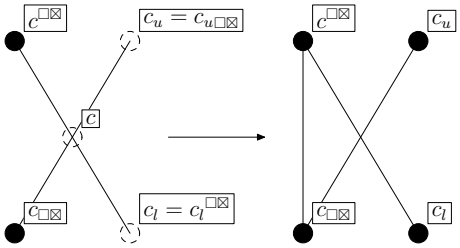
Příklad 3.28.
Příklady netriviálních transformací konceptů $\mathcal{B}(I)$ na koncepty $\mathcal{B}(J)$.



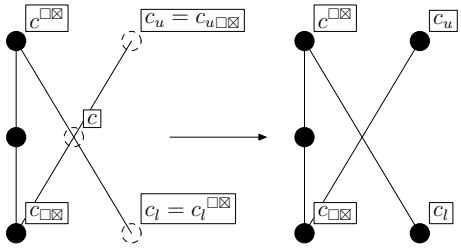
(a) Koncepty se stanou nesrovnatelnými.



(b) Prostřední nestabilní koncept se rozdělí.



(c) Prostřední nestabilní koncept zmizí.



(d) Prostřední nestabilní koncept se rozdělí. Stabilní koncept již propojuje sousedy.

V algoritmu 6 jsou použity následující funkce:

- $UpperNeighbors(c)$ - vrací horní sousedy konceptu c;
- $LowerNeighbors(c)$ - vrací dolní sousedy konceptu c;
- $Link(c_1, c_2)$ - zavede sousedský vztah mezi c_1 a c_2;
- $Unlink(c_1, c_2)$ - zruší sousedský vztah mezi c_1 a c_2.

Implementace těchto funkcí závisí na konkrétní datové reprezentaci konceptů a proto ji nebudeme uvádět a vystačíme si s uvedeným slovním popisem jejich funkčnosti.

Algoritmus 6 Transformace $\mathcal{B}(I)$, včetně informace o struktuře, na $\mathcal{B}(J)$.

procedure LinkIfNeeded($c_1, c_2 \in \mathcal{B}$)
  if $\nexists k \in \mathcal{B}(I) : c_1 < k < c_2$ **then**
    $Link(c_1, c_2)$;
  end if
end procedure

procedure SplitConcept($c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$)
  $d_1 = c_{\square}$; $d_2 = c^{\square}$;
  $Link(d_1, d_2)$;
  for all $u = \langle C, D\rangle \in UpperNeighbors(c)$ **do**
    $Link(d_2, u)$; $Unlink(c, u)$;
  end for
  for all $l = \langle C, D\rangle \in LowerNeighbors(c)$ **do**
    $Link(l, d_1)$; $Unlink(l, c)$;
  end for
  for all $u = \langle C, D\rangle \in UpperNeighbors(d_2)$ **do**
    if $u \neq u^{\square\boxtimes}$ **then**
      $Unlink(d_2, u)$; $Link(d_1, u)$; $LinkIfNeeded(d_2, u^{\square\boxtimes})$;
    end if
  end for
  for all $l = \langle C, D\rangle \in LowerNeighbors(d_1)$ **do**
    if $y_0 \notin D$ **then**
      $Unlink(l, d_1)$; $Link(l, d_2)$; $LinkIfNeeded(l_{\boxtimes\square}, d_1)$;
    end if
  end for
  return d_1, d_2;
end procedure

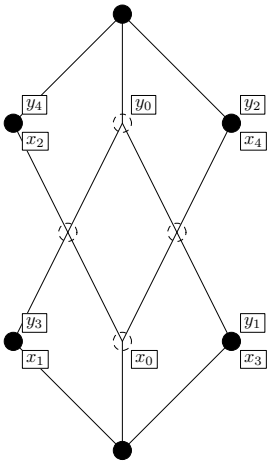
procedure RelinkReducedIntent($c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$)
  for all $u = \langle C, D\rangle \in UpperNeighbors(c)$ **do**
    if $u \neq u^{\square\boxtimes}$ **then**
      $Unlink(c, u)$;
      $LinkIfNeeded(c_{\square\boxtimes}, u)$;
      $LinkIfNeeded(c, u^{\square\boxtimes})$;
    end if
  end for
end procedure

procedure UnlinkVanishedConcept($c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$)
  for all $u = \langle C, D\rangle \in UpperNeighbors(c)$ **do**
    $Unlink(c, u)$; $LinkIfNeeded(c_{\square\boxtimes}, u)$;
  end for
  for all $l = \langle C, D\rangle \in LowerNeighbors(c)$ **do**
    $Unlink(l, c)$;
  end for
end procedure

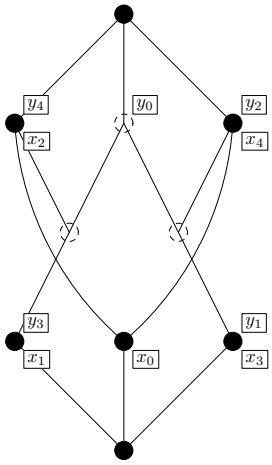
procedure TransformConceptLattice($\mathcal{B}(I)$)
  for all $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ from least to largest by levels **do**
    if $c = c^{\square\boxtimes}$ **and** $c = c_{\square\boxtimes}$ **then** ▷ Koncept se rozdělí.
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \setminus \{c\}$;
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \cup SplitConcept(c)$;
    else if $c \neq c^{\square\boxtimes}$ **and** $c = c_{\square\boxtimes}$ **then** ▷ Zmenší se extent.
      $A \leftarrow A \setminus \{x_0\}$;
    else if $c = c^{\square\boxtimes}$ **and** $c \neq c_{\square\boxtimes}$ **then** ▷ Zmenší se intent.
      $RelinkReducedIntent(c)$;
      $B \leftarrow B \setminus \{y_0\}$;
    else if $c \neq c^{\square\boxtimes}$ **and** $c \neq c_{\square\boxtimes}$ **then** ▷ Koncept nebude v $\mathcal{B}(J)$.
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \setminus \{c\}$;
      $UnlinkVanishedConcept(c)$;
    end if
  end for
end procedure

V algoritmu 6 jsou nestabilní koncepty (tvoří interval) zpracovávány po úrovních. V důsledku tohoto se nestabilní koncept zpracuje až jsou zpracovány všechny menší (vzhledem k uspořádání konceptů) nestabilní koncepty. Obecně není nutné procházet nestabilní koncepty v pořadí jak je tomu v algoritmu 6. Lze navrhnout algoritmus, který při nedeterministickém výběru konceptu pro zpracování dospěje ke korektnímu výsledku. Takový algoritmus lze jednoduše odvodit z představených výsledků, ale jeho popis je technicky náročnější a delší.

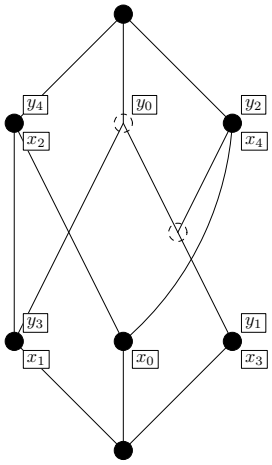
Příklad 3.29.
Běh algoritmu 6 na konceptuálním svazu příslušném formálnímu kontextu 1g. Každý obrázek zachycuje stav svazu po transformaci jednoho nestabilního konceptu.



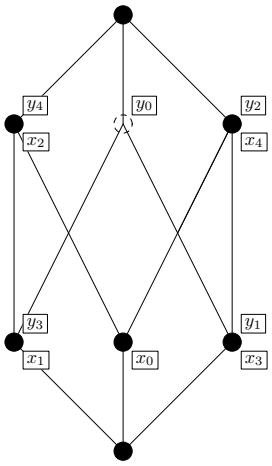
(a) Počáteční stav $\mathcal{B}(I)$.



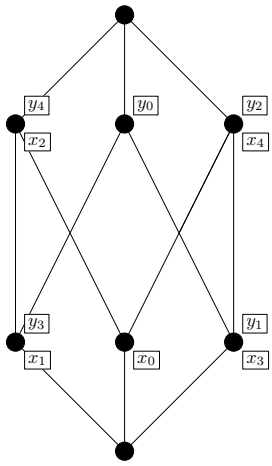
(b) Transformace konceptu $\gamma_I(x_0)$.



(c) Zmizí jeden koncept.



(d) Zmizí druhý koncept.



(e) Transformace konceptu $\mu_I(y_0)$.

Závěr

Představil jsem úvod do problematiky neúplných dat z pohledu formální konceptuální analýzy a s tím spojené vybrané metody pro práci s takovými daty. Zejména jsem se zaměřil na metody zúplňování neúplných dat. Při studiu těchto metod jsem se zároveň snažil dospět k vlastním výsledkům z této oblasti, což mě dovedlo ke zkoumání experimentální metody zúplňování pomocí počítání konceptů. Analýza této metody mě dále vedla k souvislým problémům a ve výsledku jsem se více věnoval zkoumání těchto problémů než samotnému studiu již známých metod. Celkově práce obsahuje několik výsledků týkajících se zúplnění neúplného formálního kontextu, který obsahuje právě jednu neznámou hodnotu.

Prvním výsledkem jsou experimenty se zúplňováním na základě počítání konceptů. Výsledky těchto experimentů vykazují zajímavou úspěšnost predikce neznámé hodnoty pomocí této metody. V průměru byla její úspěšnost 65 %, ve 12 % nešlo na jejím základě rozhodnout.

Podrobnější zkoumání vedlo ke druhému výsledku, což je analýza možných změn v konceptuálním svazu při odebrání incidence. Zejména se jedná o identifikaci konceptů, které jsou odebráním ovlivněny, a o analýzu změny struktury konceptuálního svazu. Za tímto účelem jsem představil čtyři operátory, které udávají určitý vztah mezi koncepty možných zúplnění. Tyto výsledky poté vedly na několik algoritmů. První z těchto algoritmů provádí výpočet počtu konceptů jednoho zúplnění ze druhého. Na tento algoritmus navazuje další, který počítá zúplnění neúplného formálního kontextu pomocí počítání konceptů. Dále je představen algoritmus, který transformuje koncepty jednoho zúplnění na koncepty druhého zúplnění. Poslední z představených algoritmů transformuje konceptuální svaz příslušný jednomu zúplnění na konceptuální svaz příslušný druhému zúplnění.

U představených algoritmů jsem se nezabýval otázkou jejich časové složitosti, ale pouze aplikací vlastních výsledků. Avšak hrubé odhady lze ve většině případů snadno určit. U všech algoritmů, vyjma posledního (transformace celého konceptuálního svazu), je nutné prozkoumat jeden celý interval (v nejhorším případě celý konceptuální svaz) a pro každý jeho prvek určit, zda je pevným bodem některého ze dvou daných uzávěrových operátorů. Z tohoto dostáváme, že asymptotická horní hranice časové složitosti těchto algoritmů nemůže být vyšší než v případě výpočtu celého konceptuálního svazu pomocí některé ze známých metod. Tento odhad je možné zpřesnit, pokud vezmeme v úvahu všechny představené výsledky jako například zjednodušení testování, zda je koncept pevným bodem některého z daných uzávěrových operátorů.

Jak již bylo zmíněno, představené výsledky se týkají neúplných formálních kontextů s právě jednou neznámou hodnotou. Konkrétně, většina z nich vychází ze znalosti zúplnění, kde je neznámá hodnota nahrazena příslušnou incidencí. Rozšíření na neúplné formální kontexty s libovolným počtem neznámých hodnot lze jednoduše, avšak naivně, realizovat. Stačí vzít výchozí zúplnění takové, že

všechny neznáme hodnoty jsou nahrazeny příslušnými incidencemi. Poté lze po jedné tyto incidence odebírat a aplikovat představené výsledky. Takovéto rozšíření je naivní kvůli jeho zjevné neefektivitě. Efektivnější rozšíření představených výsledků, stejně jako jejich přizpůsobení pro FKA s fuzzy atributy, je otázkou budoucí práce.

Conclusions

In real applications, one may need to process incomplete data. Unfortunately, most data processing methods consider only complete data as a valid input. This thesis provides an introduction to the topic of processing incomplete data by formal concept analysis. However, main results of my work are new theoretical findings in this area, considering incomplete formal contexts with exactly one unknown value.

The first result is a set of experiments with an experimental method for computing completion of incomplete context, based on counting concepts. Outcome of these experiments shows an interesting success rate of this method. In average, the success rate was around 65 %, in 12 % it was not possible to compute completion based on this method.

Further analysis of this method led to the second result, which is an analysis of all possible changes that can a concept lattice undergo after a removal of one incidence from the corresponding formal context. Based on this result, one can identify concepts that are affected by the removal and one has complete description how these changes are reflected in the structure of corresponding concept lattice. Those results were basis for several algorithms presented in this thesis. Specifically, there is an algorithm for computing the number of concepts of a completion from the other one. There is also an algorithm transforming concepts of a completion to concepts of the other completion. And finally, there is an algorithm transforming whole concept lattice.

There is no complexity analysis for presented algorithms, because their purpose is only to apply presented theoretical findings. However, one can easily get rough estimate for almost all of them (excluding the one for transformation of whole concept lattice). By an easy inspection, one can conclude that their asymptotic upper bound for the time complexity can not be higher than for computing the whole concept lattice from the scratch. In fact, the estimate could be better, if one will take in account all the results, which simplify any part of these algorithms.

As I mentioned before, my results apply to incomplete contexts with exactly one unknown value. Specifically, presented results are usually based on the knowledge of the completion with unknown value replaced by a corresponding incidence. An extension, which applies to the incomplete contexts with arbitrary number of unknown values, could be simply done in a naive way. One can take a completion, where all unknown values are replaced by corresponding incidences. Then remove these incidences one by one and apply presented findings. This kind of extension is naive, because of its obvious inefficiency. An efficient extension, as well as adaptation of these findings to FCA with fuzzy attributes, is the matter of the future research.

Reference

[1] R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In I. Rival, editor, *Ordered Sets*, pages 445–470. Boston, 1982.

[2] R. Belohlavek. Introduction to formal concept analysis. Dostupné na adrese http://phoenix.inf.upol.cz/esf/ucebni/formal.pdf, 2008.

[3] Bernard. Ganter and Rudolf. Wille. *Formal Concept Analysis – Mathematical Foundations*. Springer, 1999.

[4] Bernhard Ganter. Two basic algorithms in concept analysis. FB4–Preprint 831, TH Darmstadt, 1984.

[5] Richard Holzer. Knowledge acquisition under incomplete knowledge using methods from formal concept analysis: Part I. *Fundam. Inf.*, 63(1):17–39, 2004.

[6] Richard Holzer. Knowledge acquisition under incomplete knowledge using methods from formal concept analysis: Part II. *Fundam. Inf.*, 63(1):41–63, 2004.

[7] Peter Burmeister and Richard Holzer. On the treatment of incomplete knowledge in formal concept analysis. In Bernhard Ganter and Guy Mineau, editors, *Conceptual Structures: Logical, Linguistic, and Computational Issues*, volume 1867 of *Lecture Notes in Computer Science*, pages 385–398. Springer Berlin / Heidelberg, 2000.

[8] Sergei A. Obiedkov. Modal logic for evaluating formulas in incomplete contexts. In *Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces*, ICCS '02, pages 314–325, London, UK, UK, 2002. Springer-Verlag.

[9] Lenka Piskova, Stefan Pero, Tomas Horvat, and Stanislav Krajci. Mining concepts from incomplete datasets utilizing matrix factorization. In *CLA 2012*, 2012.

[10] K. Bache and M. Lichman. UCI machine learning repository. Dostupné na adrese http://archive.ics.uci.edu/ml, 2013.

[11] Jun Liu and Xiao qiu Yao. Formal concept analysis of incomplete information system. In Maozhen Li, Qilian Liang, Lipo Wang, and Yibin Song, editors, *FSKD*, pages 2016–2020. IEEE, 2010.

[12] Michal Krupka and Jan Lastovicka. Concept lattices of incomplete data. In *Lecture Notes in Computer Science*, volume 7278, pages 180–194, 2012.

[13] Leila Ben Othman and Sadok Ben Yahia. Yet another approach for completing missing values. In SadokBen Yahia, EngelbertMephu Nguifo, and Radim Belohlavek, editors, *Concept Lattices and Their Applications*, volume 4923 of *Lecture Notes in Computer Science*, pages 155–169. Springer Berlin Heidelberg, 2008.

[14] Rudolf Wille. Concept lattices and conceptual knowledge systems. *Computers & Mathematics with Applications*, 23(6–9):493 – 515, 1992.

[15] Radim Belohlavek and Martin Trnecka. Basic level of concepts in formal concept analysis. In Florent Domenach, DmitryI. Ignatov, and Jonas Poelmans, editors, *Formal Concept Analysis*, volume 7278 of *Lecture Notes in Computer Science*, pages 28–44. Springer Berlin Heidelberg, 2012.

[16] R. Wille and J. Stahl. Preconcepts and set representation of contexts. In *Gaul & Schader (eds): Classification as a tool of research*, 1986.

A. Obsah přiloženého CD

V samotném závěru práce je uveden stručný popis obsahu přiloženého CD/DVD, tj. závazné adresářové struktury, důležitých souborů apod.

`doc/`
: Dokumentace práce ve formátu PDF, vytvořená dle závazného stylu KI PřF pro diplomové práce, včetně všech příloh, a všechny soubory nutné pro bezproblémové vygenerování PDF souboru dokumentace, tj. zdrojový text dokumentace, vložené obrázky, apod.

`experiments/`
: Vybrané datasety, které jsem použil pro experimenty uvedené v této práci.

`readme.txt`
: Soubor s popisem obsahu přiloženého CD/DVD.

U veškerých odjinud převzatých materiálů obsažených na CD/DVD jejich zahrnutí dovolují podmínky pro jejich šíření nebo přiložený souhlas držitele copyrightu. Pro materiály, u kterých toto není splněno, je uveden jejich zdroj (webová adresa) v textu dokumentace práce.