POLSKA AKADEMIA NAUK
Instytut Badań Systemowych

ROZMYTOŚĆ I BIPOLARNOŚĆ W INTELIGENTNYM WYSZUKIWANIU INFORMACJI

Sławomir Zadrożny

Warszawa 2013

POLSKA AKADEMIA NAUK
INSTYTUT BADAŃ SYSTEMOWYCH

Seria: BADANIA SYSTEMOWE
Tom 73

Redaktor naukowy:
Prof. dr hab. inż. Jakub Gutenbaum

Warszawa 2013

Rada redakcyjna serii: BADANIA SYSTEMOWE

POLSKA AKADEMIA NAUK
INSTYTUT BADAŃ SYSTEMOWYCH

Sławomir Zadrożny

ROZMYTOŚĆ I BIPOLARNOŚĆ W INTELIGENTNYM WYSZUKIWANIU INFORMACJI

Warszawa 2013

Autorzy:

Dr hab. Sławomir Zadrożny
Instytut Badań Systemowych Polskiej Akademii Nauk
ul. Newelska 6, 01-447 Warszawa
Slawomir.Zadrozny@ibspan.waw.pl

Recenzenci:

dr hab. inż. Maciej Krawczak
dr Marek Reformat

Skład: Aneta M. Pielak

Wydawca:

Instytut Badań Systemowych
Polskiej Akademii Nauk
Newelska 6, 01-447 Warszawa
www.ibspan.waw.pl

ISSN 0208-8029
ISBN 83-894-7551-0

Rozdział 3

Relacyjne bazy danych i ich rozszerzenia

Zastosowania logiki rozmytej omawiane w niniejszej książce mają charakter ogólny i znajdują zastosowanie w ramach różnych modeli danych. Jednak model relacyjny, ze względu na swoją popularność i solidne podstawy matematyczne, stanowi zwykle oryginalne środowisko, w którym te zastosowania zostały zaproponowane. W związku z tym, również w niniejszej książce stanowi on podstawę rozważań i jego aparat pojęciowy zostanie pokrótce omówiony w tym rozdziale.

3.1 Klasyczny relacyjny model danych

W ujęciu abstrakcyjnym *relacyjna baza danych* może być postrzegana jako zbiór *relacji* [64]. Zasadniczo należy ją rozumieć jako *relację matematyczną*, czyli podzbiór iloczynu kartezjańskiego ustalonych zbiorów. W rozważaniach dotyczących baz danych dogodnie jest odróżnić pojęcie *schematu* (*nagłówka*) relacji i *instancji* (*treści*) relacji. Schemat relacji określa jakie zbiory – zwane *dziedzinami* relacji – występują w rozważanym iloczynie kartezjańskim. Dodatkowo schemat relacji wiąże z każdą z jej dziedzin pojęcie *atrybutu* relacji. Schemat relacji R można więc przedstawić w następującej postaci

$$\{A_1 : D_1, \ldots, A_n : D_n\} \tag{3.1}$$

gdzie A_i jest jej i-tym atrybutem i D_i jest dziedziną (typem danych) i-tego atrybutu, oznaczaną również jako dom_{A_i}. Jeśli jawne określenie dziedzin poszczególnych atrybutów nie jest istotne, to schemat relacji

w skrócie zapisuje się jako $\{A_1, \ldots, A_n\}$. Dziedzina D_i określa jakie wartości może przyjmować atrybut A_i i jakie operacje można na nich wykonywać. Każda relacja reprezentuje zbiór bytów istotnych dla modelowanego wycinka świata rzeczywistego. Każdy byt reprezentowany jest przez *krotkę* relacji, oznaczaną jako t:

$$t \in D_1 \times D_2 \times \ldots \times D_n, \qquad t = (a_1, a_2, \ldots, a_n), \qquad a_i \in D_i$$

Instancją relacji nazywa się jej *zawartość*, czyli pewien zbiór krotek.

Poszczególne dziedziny D_i mogą być zbiorami liczb całkowitych bądź rzeczywistych, łańcuchów znaków czy dat. Klasycznie przyjmuje się, że wartości poszczególnych dziedzin są w pewnym sensie *atomowe* (*niepodzielne*), czyli nie posiadają wewnętrznej struktury. Odejście od tego wymogu jest jednym z podstawowych cech *obiektowego* czy *relacyjno-obiektowego modelu danych* (por. np. [69]), jak i różnych podejść związanych z reprezentacją *informacji niepewnej* w bazie danych [178].

Model relacyjny przewiduje prosty sposób reprezentacji informacji niekompletnej: w sytuacji, kiedy wartość atrybutu w krotce jest nieznana lub z innych powodów nieokreślona [75, 178] oznacza się to za pomocą pseudowartości `null`.

Przyjmuje się, że istnieje taki zestaw atrybutów, że łączne podanie ich wartości jednoznacznie identyfikuje krotkę i żaden podzbiór tego zestawu juz tej własności nie posiada. Taki zestaw atrybutów nazywa się *kluczem* relacji.

W mniej formalnym ujęciu relacje są traktowane jako *tabele*. Kolumny i wiersze tabeli odpowiadają atrybutom relacji i ich krotkom.

Tak więc dane przechowywane są w *bazie danych* i udostępniane użytkownikom za pośrednictwem *systemu zarządzania bazą danych*. Specyfika takich systemów skonstruowanych zgodnie z relacyjnym modelem danych polega na tym, że dane z punktu widzenia użytkownika przechowywane są wyłącznie w tabelach i wszelkie operacje jakie się na danych wykonuje odnoszą się do całych zbiorów wierszy w tych tabelach i dają w wyniku również tabele. Takie systemy nazywamy *relacyjnymi systemami zarządzania bazą danych* lub krócej *relacyjnymi bazami danych*.

Tradycyjnie wyróżnia się operacje na tabelach (relacjach), polegające na ich tworzeniu, modyfikowaniu i usuwaniu, oraz operacje na danych zawartych w tabelach. Do podstawowych spośród tych ostatnich zaliczyć należy: wstawianie wierszy, usuwanie wierszy, aktualizowanie danych oraz wyszukiwanie danych.

Szczególnie interesująca jest *operacja wyszukiwania danych*. Zazwyczaj w tym celu użytkownik formułuje *zapytanie*, które określa zakres

poszukiwanych danych i warunki jakie powinny one spełniać. Wyróżnia się dwa typy *języków zapytań* do relacyjnych baz danych. Pierwszy typ ma charakter *deklaratywny* i odwołuje się do *rachunku relacyjnego* – specjalnej wersji *rachunku predykatów pierwszego rzędu*. W tym przypadku zapytanie stanowi jedynie sformalizowany zapis kryteriów jakie powinny spełniać poszukiwane dane, natomiast określenie sposobu wyszukania danych całkowicie pozostawia się systemowi zarządzania bazą danych. Drugi typ języków zapytań ma charakter *proceduralny* i oparty jest na *algebrze relacji*. Zapytanie określa nie tylko kryteria, ale również sekwencję działań na relacjach, które należy zrealizować, aby skonstruować relację wynikową zawierającą poszukiwane dane. Algebra relacji obejmuje pięć podstawowych operacji: *sumę*, *różnicę*, *iloczyn kartezjański*, *rzut* i *wybór* . Ich złożenia pozwalają zdefiniować pozostałe użyteczne operacje, takie jak *przecięcie*, *dzielenie* czy *złączenie*. Definicje poszczególnych operacji podane są w p. 3.1.1. Z rozważań teoretycznych wynika, że obydwa podejścia są równoważne w tym sensie, że można z użyciem każdego z nich uzyskać opis dowolnego podzbioru danych zgromadzonych w bazie. W komercyjnych implementacjach relacyjnych systemów zarządzania bazą danych dominuje aktualnie *język SQL* będący do pewnego stopnia połączeniem obydwu tych podejść. Język SQL jest również często punktem wyjścia badań zmierzających do jego rozszerzenia.

Dla ilustracji niektórych zagadnień posłużymy się w książce przykładem tabeli `NIERUCHOMOSCI` z bazy danych ofert hipotetycznej agencji nieruchomości. Jej struktura przedstawiona jest w tabl. 3.1.

Kolumna	Typ danych	Opis
`id`	`NUMERIC(5)`	*identyfikator nieruchomości*
`adres`	`CHAR(50)`	*miejscowość*
`cena`	`NUMERIC(10,2)`	*cena nieruchomości w złotych*
`powierzchnia`	`NUMERIC(10,2)`	*powierzchnia mieszkalna*
`lokalizacja`	`CHAR(10)`	*region/dzielnica gdzie się znajduje*
`od_stacji`	`NUMERIC(4)`	*odległość od stacji kolejowej*

Tablica 3.1: Struktura przykładowej tabeli `NIERUCHOMOSCI`

3.1.1 Algebra relacji i rachunek relacyjny

Algebra relacji

Operacje algebry relacji przyjmują jako argument jedną lub dwie relacje i w wyniku dają relację. Opiszemy teraz poszczególne operacje, stosując przy tym następującą notację: R i S oznaczają relacje, t oznacza krotkę, zaś $t \in R$ oznacza, że t jest krotką relacji R. Pięć podstawowych operacji definiuje się następująco:

suma $(R \cup S)$

$$t \in R \cup S \Leftrightarrow (t \in R) \vee (t \in S) \tag{3.2}$$

różnica $(R \setminus S)$

$$t \in R \setminus S \Leftrightarrow (t \in R) \wedge (t \notin S) \tag{3.3}$$

iloczyn kartezjański $(R \times S)$

$$(a_1, \ldots, a_n, b_1, \ldots, b_m) \in R \times S \Leftrightarrow \\ \Leftrightarrow (a_1, \ldots, a_n) \in R \ \wedge \ (b_1, \ldots, b_m) \in S \tag{3.4}$$

rzut $(\pi_{A_1,\ldots,A_k}(R))$

$$(a_1, \ldots, a_k) \in \pi_{A_1,\ldots,A_k}(R) \Leftrightarrow \ \exists_{a_{k+1},\ldots,a_n} \\ (a_1, \ldots, a_k, a_{k+1}, \ldots, a_n) \in R \qquad a_i \in D_i \tag{3.5}$$

wybór $(\sigma_W(R))$

$$(a_1, \ldots, a_n) \ \in \ \sigma_W(R) \ \Leftrightarrow \ (a_1, \ldots, a_n) \ \in \ R \wedge W(a_1, \ldots, a_n) \tag{3.6}$$

W wypadku dwóch pierwszych operacji zakłada się, że schematy relacji R i S są zgodne. W wypadku iloczynu kartezjańskiego zakłada się, że nazwy atrybutów w obu relacjach są różne.

Trzy pierwsze operacje mają oczywistą interpretację teoriomnogościową. Iloczyn kartezjański jest nieco zmodyfikowany. Formalnie w wyniku powinno otrzymać się zbiór par uporządkowanych elementów z relacji R i S o schematach, odpowiednio $\{A_1, \ldots, A_n\}$ i $\{B_1, \ldots, B_m\}$. Zamiast tego otrzymuje się zbiór krotek relacji o schemacie:

$$\{A_1, \ldots, A_n, B_1, \ldots, B_m\}$$

W operacji rzutu schemat relacji wynikowej jest podzbiorem schematu relacji wejściowej R, a każda krotka R daje w wyniku krotkę relacji

wynikowej (o odpowiednio zredukowanej liczbie atrybutów). Zawartość relacji wynikowej poddawana jest jeszcze usuwaniu duplikatów, czyli krotek o identycznych wartościach wszystkich atrybutów.

W operacji wyboru warunek W skonstruowany jest z użyciem atrybutów relacji R, stałych właściwych dla dziedzin tych atrybutów oraz takich operatorów porównania jak "$=$" czy "$\geq$".

Operacja *przemianowania* jest również często zaliczana do podstawowych operacji algebry relacji:

przemianowanie $(\rho_{\{(A_1,B_1),\ldots,(A_k,B_k)\}}(R))$

$$(B_1 : a_1, \ldots, B_k : a_k, A_{k+1} : a_{k+1}, \ldots, A_n : a_n) \in S \Leftrightarrow (A_1 : a_1, \ldots, A_n : a_n) \in R \quad (3.7)$$

gdzie $S = \rho_{\{(A_1,B_1),\ldots,(A_k,B_k)\}}(R)$ i dla atrybutów krotek podajemy nie tylko ich wartości, ale również nazwy.

Operacja ta ma raczej techniczny charakter. W wyniku jej działania otrzymuje się nową relację o zawartości identycznej z oryginalną i schemacie różniącym się jedynie nazwami atrybutów. Przy złożeniu z inną operacją, taką jak na przykład iloczyn kartezjański, pozwala to spełnić wymagania dotyczące unikalności nazw atrybutów w relacjach będących ich argumentami.

Wymienimy jeszcze kilka użytecznych operacji, które są jednak wyrażalne w postaci złożenia podstawowych operacji algebry relacji. Zaliczają się do nich:

przecięcie $(R \cap S)$

$$t \in R \cap S \Leftrightarrow (t \in R) \wedge (t \in S), \quad (3.8)$$

które można wyrazić z użyciem operacji różnicy:

$$R \cap S = R \setminus (R \setminus S) \quad (3.9)$$

złączenie $(R \bowtie_W S)$

$$R \bowtie_W S = \sigma_W(R \times S) \quad (3.10)$$

dzielenie $(R \div S)$

$$(a_1, \ldots, a_k) \in R \div S \Leftrightarrow \forall_{(a_{k+1},\ldots,a_n)\in S}(a_1, \ldots, a_k, a_{k+1}, \ldots, a_n) \in R, \quad (3.11)$$

które można wyrazić w następujący sposób:

$$R \div S = \pi_{A_1,\ldots,A_k}(R) \setminus \pi_{A_1,\ldots,A_k}((\pi_{A_1,\ldots,A_k}(R) \times S) \setminus R), \qquad (3.12)$$

przy czym schemat relacji S jest podzbiorem schematu relacji R.

Operacja *złączenia* stanowi złożenie dwóch operacji: iloczynu kartezjańskiego i wyboru. Stosuje się ją w celu utworzenia relacji, której krotki zawierają dane pochodzące z dwóch, powiązanych ze sobą warunkiem W, krotek relacji R i S.

Przy operacji *dzielenia* zakłada się, że relacje R i S, będące jej argumentami, mają schematy odpowiednio postaci $\{A_1, \ldots, A_k, A_{k+1} \ldots, A_n\}$ i $\{A_{k+1}, \ldots, A_n\}$. Tak więc schemat relacji S jest podzbiorem schematu relacji R.

Niektórzy autorzy wprowadzają dodatkowe operacje algebry, w szczególności odpowiedniki operacji agregacji znanych z języka SQL.

Rachunek relacyjny

Istota *rachunku relacyjnego* polega na użyciu pewnej wersji *rachunku predykatów pierwszego rodzaju* do opisu poszukiwanych danych. Rachunek relacyjny występuje w literaturze w dwóch wariantach: *rachunku dziedzin* i *rachunku krotek*. Omówimy teraz pokrótce pierwszy z tych wariantów.

Formuły logiczne rachunku predykatów interpretuje się względem schematu rozważanej bazy danych. Symbole zmiennych $x_1, x_2, \ldots$, nazywane *zmiennymi dziedzinowymi*, znajdują swoją interpretację jako elementy dziedzin atrybutów występujących w schematach relacji zdefiniowanych w bazie danych. Symbole predykatów $R, S, \ldots$ interpretuje się jako relacje w rozważanej bazie danych. Zapytania przyjmują następującą postać:

$$\{x_1, \ldots x_n \mid \phi(x_1, \ldots x_m)\} \qquad m \geq n \qquad (3.13)$$

gdzie ϕ jest formułą. Odpowiedzią na to zapytanie jest relacja, której krotki *spełniają* formułę ϕ. Formuła ta może przyjmować następującą postać:

1. formuły atomowej, czyli
   - $R(x_1, \ldots, x_n)$, gdzie R oznacza relację o n atrybutach,
   - $x_1 \theta x_2$, gdzie θ jest operatorem porównania, takim jak "$=$", "$<$",. . . ,
   - $x_1 \theta c$, gdzie c jest *stałą*,

2. $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi$, gdzie ϕ, ϕ_1, ϕ_2 są formułami, zaś "$\wedge$", "$\vee$", "$\neg$" oznaczają odpowiednio koniunkcję, alternatywę i negację,

3. $\exists x \phi(x)$ oraz $\forall x \phi(x)$, gdzie ϕ jest formułą, zaś x zmienną wolną w niej występującą.

Zbiór krotek stanowiących odpowiedź na zapytanie (3.13) przy powyżej określonej postaci formuły ϕ może być nieskończony. Przykładem takiego zapytania może być:

$$\{x \mid \neg\texttt{NIERUCHOMOSCI}(1, x, 'Warszawa', 250, 'Mazowsze', 300)\},$$

gdzie relacja NIERUCHOMOSCI określona jest jak w tab. 3.1. W celu uniknięcia tej niepożądanej sytuacji dopuszcza się używanie wyłącznie tak zwanych *bezpiecznych formuł* (por. np. [213]).

Początkowo rachunek relacyjny kandydował do roli standardowego języka zapytań dla systemów opartych na relacyjnym modelu danych. Obecnie znajduje zastosowanie głównie w rozważaniach o charakterze akademickim. Pozostaje jednak interesującym narzędziem do analizy języków zapytań.

3.1.2 Język SQL

Język SQL (ang. *Structured Query Language*) jest aktualnie standardowym językiem relacyjnych systemów zarządzania bazą danych. Z punktu widzenia wyszukiwania danych interesować nas będzie wyłącznie instrukcja SELECT języka SQL. Ze względu na swą podstawową funkcję określana ona bywa mianem *zapytania* (ang. *query*). Jej składnię można w uproszczony sposób przedstawić następująco:

SELECT	*lista wyrażeń*
FROM	*lista tabel*
WHERE	*warunek*
GROUP BY	*lista wyrażeń grupujących*
HAVING	*warunek dotyczący grup wierszy*
ORDER BY	*lista wyrażeń porządkujących*

(3.14)

gdzie wielkimi literami przedstawiono słowa kluczowe języka SQL, zaś napisy podane małymi literami określają w sposób opisowy, co powinno być wstawione w ich miejsce. W szczególności wyrażenia konstruuje się w języku SQL, podobnie jak w innych językach programowania, z użyciem: stałych, nazw kolumn (odpowiadają tu one zmiennym), operatorów arytmetycznych i różnych funkcji standardowych pozwalających

operować na danych poszczególnych typów. Szczególną rolę odgrywają *funkcje agregujące*, które zwracają pojedynczą wartość dla całych grup wierszy. Poszczególne fragmenty instrukcji, pokazane w kolejnych wierszach (3.14), nazywamy *frazami*. W szczególności fragment rozpoczynający się od słowa kluczowego `SELECT` nazywamy "frazą `SELECT`", fragment rozpoczynający się od słowa kluczowego `FROM` "frazą `FROM`" i tak dalej.

Wynikiem wykonania instrukcji `SELECT` jest nowa tabela (relacja) o następujących cechach:

- ma tyle kolumn, ile podano wyrażeń na liście we frazie `SELECT`; poszczególne kolumny zawierają dane będące wartościami tych wyrażeń obliczonymi dla pojedynczych wierszy lub ich grup,
- dane w niej zawarte obliczane są na podstawie danych pochodzących z tabeli lub złączenia tabel wymienionych we frazie `FROM` i spełniających warunek występujący we frazie `WHERE`,
- przed obliczeniem wyrażeń występujących we frazie `SELECT` wiersze mogą być pogrupowane według wartości wyrażeń grupujących wymienionych we frazie `GROUP BY`; dodatkowo, posługując się frazą `HAVING` można narzucić pewne warunki na uzyskane grupy, tak że tylko grupy je spełniające są uwzględnione przy określaniu zawartości tabeli wynikowej całej instrukcji `SELECT` (fraza `HAVING` odgrywa więc tę samą rolę wobec grup wierszy co fraza `WHERE` wobec pojedynczych wierszy zanim zostaną one ewentualnie pogrupowane),
- fraza `ORDER BY` umożliwia określenie kolejności wierszy w tabeli wynikowej - są one posortowane zgodnie z wartościami wyrażeń porządkujących.

Istnieje możliwość zagnieżdżania instrukcji `SELECT` w innych instrukcjach języka SQL. Zagnieżdżoną instrukcję `SELECT` nazywać będziemy *podzapytaniem*. Tabela będąca wynikiem zagnieżdżonej instrukcji `SELECT` interpretowana jest odpowiednio do kontekstu w którym występuje. Na przykład, wynik podzapytania występujacego we frazie `FROM` interpretowany jest wprost jako tabela, natomiast wynik podzapytania występujacego jako część wyrażenia (przy spełnieniu odpowiednich warunków) interpretowany jest jako wielkość skalarna. Istnieją również specjalne konstrukcje w ramach frazy `WHERE` instrukcji `SELECT`, których używa się przede wszystkim lub wyłącznie wraz z podzapytaniem. Należą do nich operatory `IN`, `EXISTS`, `ANY` i `ALL`.

Instrukcje `SELECT` można łączyć ze sobą również w inny sposób: z użyciem operacji teoriomnogościowych takich jak: suma, przecięcie i różnica, wyrażanych z użyciem słów kluczowych `UNION`, `INTERSECT` i `EXCEPT`. Interpretacja tych operacji jest oczywista, jeśli pamięta się, że wynikiem instrukcji `SELECT` jest tabela, czyli zbiór wierszy. Schematy (nagłówki) łączonych w ten sposób tabel muszą być zgodne.

3.2 Modelowanie informacji niedoskonałej i rozmyte rozszerzenia modelu relacyjnego

3.2.1 Informacja niekompletna i pseudowartość `null`

Model relacyjny w ograniczonym zakresie pozwala reprezentować informację nieprecyzyjną czy niepewną. Praktycznie jedynym narzędziem temu służącym jest *pseudowartość*[1] `null`, która pozwala wyrazić brak informacji o wartości danego atrybutu relacji dla danej krotki [65, 66, 219, 15, 259]. Stosowanie pseudowartości `null` ma jednak wady.

Po pierwsze, brak jej jednoznacznej interpretacji. Często stosuje się ją w praktyce zarówno w sytuacji, kiedy dany atrybut faktycznie posiada wartość, ale jest ona nieznana (w danej chwili), jak również wtedy kiedy dany atrybut nie stosuje się w wypadku danej krotki (np. dany klient ma wpisaną pseudowartość `null` jako wartość atrybutu `numer_faxu`, ponieważ nie posiada faxu[2]).

Po drugie, uwzględnienie pseudowartości `null` powoduje konieczność zastosowania *logiki trójwartościowej*. Jeśli wartość atrybutu A dla krotki t, oznaczona jako $A(t)$, nie jest znana, to na przykład dla warunku:

$$A(t) = x \tag{3.15}$$

gdzie $x \in dom(A)$ nie można ani stwierdzić, że jest on spełniony (ma wartość logiczną *Prawda*) ani, że nie jest spełniony (ma wartość logiczną *Fałsz*). W związku z tym wprowadza się trzecią wartość logiczną, często oznaczaną jako *Unk* (ang. *Unknown*). Przyjmują je te warunki logiczne, dla których nie można jednoznacznie określić ich wartości logicznej -

[1] Używa się określenia *pseudowartość* z tego względu, że `null` nie jest wartością atrybutu, nie należy do jego dziedziny. Stanowi jedynie *znacznik*, wskazujący że nie jest znana lub nie istnieje wartość atrybutu w danej krotce.

[2] W tym wypadku można problem przezwyciężyć zmieniając schemat relacji i umieszczając atrybut `numer_faxu` w innej, powiązanej relacji. Jednak często jest to niewygodne.

tak jak ma to miejsce w wypadku (3.15). W konsekwencji na przykład następujący warunek:

$$(A(t) = x) \vee (A(t) \neq x) \tag{3.16}$$

również przyjmuje wartość logiczną *Unk*, przy założeniu że wartość atrybutu A dla krotki t oznaczona jest z użyciem pseudowartości `null`. Wydaje się to naruszać zdroworozsądkową intuicję, skodyfikowaną w logice klasycznej jako *prawo wyłączonego środka*: dla dowolnego zdania, albo ono jest prawdziwe, albo jego zaprzeczenie jest prawdziwe. Ta niespójność w interpretacji warunku takiego jak (3.15) wynika z występującego tu pomieszania dwóch odrębnych paradygmatów: aparatu logiki klasycznej z ekstensjonalnymi[3]spójnikami logicznymi oraz wnioskowania w warunkach niepewności, którego operacje nie są ekstensjonalne (por. np. [99]). "Trzecia wartość logiczna" *Unk* nie powinna więc być traktowana jako wartość logiczna, a jedynie jako reprezentacja faktu, że wartość logiczna danego wyrażenia jest nieznana – chociaż nadal jest to *Prawda* bądź *Fałsz*. Podejście oparte na logice trójwartościowej jest jednak efektywne obliczeniowo, co stanowi pewne uzasadnienie jego zastosowania w modelu relacyjnym. Należy jednak wspomnieć, że nieobowiązywanie aksjomatów logiki klasycznej ma konsekwencje dalej idące niż tylko nieoczekiwane wartości logiczne takich warunków jak (3.16). Aksjomaty logiczne takie jak prawo wyłączonego środka, odgrywają ważną rolę w optymalizacji wykonywania zapytań. Optymlizator, element systemu zarządzania relacyjną bazą danych, dokonuje pewnych przekształceń oryginalnej postaci zapytań do postaci równoważnych w celu określenia jak najbardziej efektywnego planu ich wykonania. Nieobowiązywanie pewnych reguł logiki klasycznej ogranicza repertuar dopuszczalnych przekształceń i ogranicza skuteczność działania optymalizatora.

Codd[4] proponował w swoich pracach [66, 67, 68] stosowanie dwóch symboli na oznaczenie pseudowartości `null`, różniących się semantyką zgodnie z opisanym wcześniej rozróżnieniem: wartość atrybutu *istnieje, ale nie jest znana* oraz wartość atrybutu dla danej krotki *w ogóle nie istnieje*. W pierwszym wypadku będziemy używać oznaczenia $\texttt{null}_{\exists}$, zaś w drugim $\texttt{null}_{\perp}$. System reprezentacji informacji niekompletnej z użyciem klasycznego pojęcia pseudowartości `null` określany jest mianem *Codd tables*, por. np. [117] i def. 3.2.

Imieliński i Lipski [124] zaproponowali rozszerzenie repertuaru pseudowartości `null`, tak aby umożliwić pełniejsze wyrażenie posiadanej in-

[3] ang. *truth-functional*

[4] Twórca modelu relacyjnego.

Krotka	Atrybuty			Formuła
	A_1	A_2	A_3	
t_1	1	2	x	
t_2	3	x	y	$x = y \wedge z \neq 2$
t_3	z	4	5	$x \neq 1 \vee x \neq y$

Tablica 3.2: Przykład *c-table* [117]

formacji. Możemy w danej chwili nie posiadać informacji o adresach pracowników X i Y, ale wiemy że są małżenstwem i wspólnie zamieszkują. Przypisanie tradycyjnej pseudowartości $\texttt{null}_{\exists}$ atrybutowi `adres` dla obydwu krotek reprezentujących X i Y, oznaczonych dalej jako t_X i t_Y, nie oddaje w pełni posiadanej informacji, gdyż warunek:

$$t_X(\texttt{adres}) = t_Y(\texttt{adres}) \tag{3.17}$$

nie będzie przez system uznany za spełniony. W związku z tym proponuje się reprezentowanie nieznanych wartości atrybutów z użyciem zmiennych, przy czym jeśli $t_X(\texttt{adres})$ i $t_Y(\texttt{adres})$ przypisana jest ta sama zmienna, to warunek (3.17) jest oczywiście spełniony, natomiast jeśli są im przypisane różne zmienne, to warunek może, ale nie musi być spełniony. Taki model reprezentacji informacji niekompletnej określany jest mianem *v-tables*.

Kolejny ważny model reprezentacji informacji niedoskonałej to tak zwane *c-tables*, również zaproponowane przez Imielińskiego i Lipskiego [124]. Model ten bazuje na *v-tables*, z tym że z każdą krotką może być dodatkowo związana formuła logiczna (warunek), w której występują zmienne i stałe. Zmienne mogą być zarówno wcześniej wspomnianymi zmiennymi reprezentującymi nieznane wartości atrybutów, jak i zmiennymi używanymi wyłącznie w tych formułach. Zmienne te pozwalają narzucić dodatkowe ograniczenia na nieznane wartości atrybutów. Dodatkowo, pozwalają one uzależnić występowanie danej krotki od występowania innych krotek lub przyjmowanych przez ich atrybuty określonych wartości. Przykład relacji zapisanej zgodnie z modelem *c-tables* przedstawia tab. 3.2.1. Aby precyzyjnie opisać interpretację tej relacji wprowadzimy najpierw za [117] formalne pojęcie *bazy danych z niekompletną informacją* oraz *systemu reprezentacji* (ang. *representation system*) dla takiej bazy.

Rozważmy relacyjną bazę danych o schemacie $\{A_1 : D_1, \ldots A_n : D_n\}$, przy czym każda z dziedzin D_i jest zbiorem przeliczalnym; por. (3.1).

Zbiór wszystkich możliwych instancji I relacji o takim schemacie będziemy oznaczać $\mathcal{N}$:

$$\mathcal{N} = \{I \mid I \subseteq D_1 \times \ldots \times D_n, \quad I \text{ jest skończone}\}$$

przy czym rozważamy tylko instancje, w których wszystkie atrybuty we wszystkich krotkach mają określoną wartość należącą do odpowiedniej dziedziny D_i (pseudowartość `null` nie jest dopuszczalna). W dalszym ciągu będziemy zakładać, że baza danych zawiera tylko jedną relację, ale nie ogranicza to ogólności rozważań.

Definicja 3.1 ([117])**.** Bazą danych z niekompletną informacją IDB nazywamy dowolny zbiór instancji:

$$IDB \subseteq \mathcal{N}$$

Klasyczna relacyjna baza danych[5] jest wtedy specjalnym przypadkiem IDB, takim że $IDB = \{I\}$ dla pewnej instancji I.

Intencją takiej definicji bazy danych z niekompletną informacją jest wyrażenie semantyki pseudowartości `null` i jej rozszerzeń. Mianowicie, jeśli w klasycznej relacji dopuszczamy wystąpienie pseudowartości $\texttt{null}_{\exists}$, to taką relację można utożsamić ze zbiorem, w ogólności nieskończonym, wszystkich jej instancji, w których w miejscu pseudowartości $\texttt{null}_{\exists}$ występują kolejne wartości pochodzące z dziedziny danego atrybutu.

Ze względu na to, że IDB jest w ogólności nieskończonym zbiorem instancji I, potrzebna jest dogodna forma jej reprezentacji. Tę role pełni właśnie *system reprezentacji*.

Definicja 3.2 ([117])**.** Na system reprezentacji składa się zbiór elementów Tb nazywanych tabelami oraz funkcja (reguła) Mod, która przypisuje każdej tabeli bazę danych z niekompletną informacją $IDB = Mod(Tb)$.

Wspomniane wcześniej modele: *Codd tables*, *v-tables* i *c-tables* są przykładami systemów reprezentacji. W pierwszym modelu, Tb to relacja, w której atrybutom krotek mogą być przypisane pseudowartości `null`. Funkcja Mod przypisuje takiej relacji Tb bazę IDB stanowiącą zbiór takich kopii Tb, w których wszystkie przypisania pseudowartości `null` do $t_i(A_j)$ są zastąpione przypisaniem konkretnej wartości należącej do dom_{A_j}. Funkcję Mod można utożsamić ze zbiorem funkcji ζ, takich że:

$$\zeta : Var(Tb) \to D \tag{3.18}$$

[5]Zgodnie z przyjętym założeniem zawiera ona tylko jedną relację.

gdzie $Var(Tb)$ to zbiór pseudowartosci `null` występujących w relacji Tb, $D = \sum_i dom_{A_i}$ i $\zeta(\texttt{null}_{ij}) \in D_j$ jeśli $t_i(A_j) = \texttt{null}_{ij}$. Każda taka funkcja ζ określa nową instancję należącą do IDB, która jest identyczna z Tb za wyjątkiem tego, że wszystkie pseudowartości `null` są zastąpione konkretnymi wartościami z dziedzin odpowiednich atrybutów, jak to objaśniono wcześniej.

W drugim modelu, Tb to relacja, w której zamiast pseudowartości `null` mogą wystąpić zmienne i funkcja Mod zdefiniowana jest analogicznie jak dla *Codd tables*, z tym że różnym wystąpieniom tej samej zmiennej w Tb w tej samej instancji należącej do $Mod(Tb)$ musi być przypisana ta sama wartość. Innymi słowy, model *Codd tables* to specjalny przypadek modelu *v-tables*, w którym wszystkie zmienne występujące w Tb są od siebie różne.

Wreszcie w trzecim modelu, Tb ma postać taką samą jak w modelu *v-tables* i dodatkowo z każdą krotką może być skojarzona formuła. Funkcję Mod nadal będziemy utożsamiać ze zbiorem przyporządkowań ζ (3.18), przy czym należy pamiętać, że zmienne występują również w formułach skojarzonych z krotkami. Przy tworzeniu instancji w wynikowej IDB dla danego przyporządkowania ζ sprawdza się czy formuły związane z krotkami są spełnione po podstawieniu wartości pod zmienne zgodnie z przyporządkowaniem ζ. Te krotki, dla których formuły te nie są spełnione nie występują w instancji odpowiadającej danemu przyporządkowaniu ζ.

Wróćmy do przykładu pokazanego w tab. 3.2.1. Rozważmy za [117] trzy przykładowe wartościowania ζ i odpowiadające im instancje I należące do bazy danych z niekompletną informacją IDB reprezentowaną przez *c-table* pokazaną w tab. 3.2.1. Przykład ten potwierdza, że *c-table* może reprezentować zbiór instancji *o różnych wartościach atrybutów*, ale również *o różnej liczbie krotek*. Inaczej mówiąc, model *c-tables* pozwala na reprezentację niekompletnej informacji co do wartości poszczególnych atrybutów dla poszczególnych krotek, jak również niekompletnej informacji co do występowania bądź niewystępowania poszczególnych krotek w relacji.

W [200] rozważa się jeszcze inne systemy reprezentacji, w szczególności przy założeniu, że rozważa się wyłącznie skończone bazy danych z niepełną informacją, czyli dziedziny atrybutów dla których wartości nie są jednoznacznie określone są skończone.

Inne rozszerzenie pojęcia pseudowartości `null` pozostaje w bezpośrednim związku z rozwiązaniami zaproponowanymi w zakresie reprezentacji informacji niedoskonałej z użyciem logiki rozmytej (por. p. 3.2.2).

Przyporządkowanie ζ	Zmienne			Wynikowa instancja I
	x	y	z	
ζ_1	1	1	1	1 2 1 3 1 1
ζ_2	2	3	1	1 2 2 1 4 5
ζ_3	2	2	3	1 2 2 3 2 2 3 4 5

Tablica 3.3: Przykład trzech instancji IDB wygenerowanych na podstawie *c-table* pokazanej w tab. 3.2.1 [117]

W oryginalnym podejściu, przypisanie atrybutowi A w krotce t pseudowartości $\texttt{null}_\exists$ oznacza, że wszystko co wiemy o $A(t)$ to to, że należy ona do dziedziny tego atrybutu: $A(t) \in dom_A$. W podejściu rozszerzonym [154, 123] proponuje się użycie tak zwanych *lub-zbiorów* (ang. *disjunctive sets, or-sets*), które określają pewien podzbiór dziedziny do którego należy wartość atrybutu. Na przykład, dokładny adres zamieszkania pracownika reprezentowanego przez krotkę t może być nieznany, ale wiadomo, że jest to Warszawa, Poznań lub Wrocław[6]. Wtedy:

$$\texttt{adres}(t) = \{\texttt{Warszawa}, \texttt{Poznań}, \texttt{Wrocław}\} \tag{3.19}$$

lub, w ogólności:

$$A(t) \subseteq dom_A$$

Dla uzyskania jednolitej reprezentacji wszystkich wartości atrybutów w bazie danych mozna przyjąć, że znane wartości atrybutów reprezentowane są przez zbiory jednoelementowe: jeśli oryginalnie $A(t) = v$ to przy ujednoliceniue reprezentacji $A(t) = \{v\}$.

3.2.2 Rozmyte modele danych niedoskonałych

W p. 3.2.1 rozważamy sposoby modelowania niekompletnej informacji w ramach klasycznego modelu relacyjnego i jego rozszerzeń. W ramach

[6]Przyjmujemy, ze dziedziną atrybutu `adres` jest zbiór nazw miast.

prac prowadzonych nad zastosowaniem logiki rozmytej do modelowania szerzej rozumianej *niedoskonałości informacji*, analizuje się wiele możliwych źródeł takiej niedoskonałości; por. np. [203]. Rozważmy relację (bazę danych) o kolekcji obrazów, w której schemacie występują między innymi takie atrybuty jak: `tytuł`, `artysta`, `data_powstania`, `miejsce_powstania` i `cena`. Wśród ważnych form niedoskonałości dostępnej infomacji należy wymienić:

- *nieprecyzyjność* występującą na przykład wtedy, gdy obraz jest datowany na "początek czternastego wieku",
- *brak wiarygodności* występujący na przykład wtedy, gdy datowanie obrazu pochodzi od eksperta nie cieszącego się najwyższym uznaniem,
- *niejednoznaczność* występującą na przykład wtedy, gdy znana jest tylko nazwa miasta, w którym obraz powstał i miasta o takiej samej nazwie występują w wielu krajach czy regionach,
- *sprzeczność* występującą na przykład wtedy, gdy różne źródła podają różne tytuły obrazu,
- *niekompletność*, omawianą już w p. 3.2.1, występującą na przykład wtedy, gdy brak informacji o autorze lub gdy wiadomo jedynie, że był to Rubens lub van Dyck.

Te różne formy są ze sobą powiązane i mogą też występować jednocześnie. Z punktu widzenia reprezentacji danych mogą być one postrzegane przede wszystkim jako źródło *niepewności* co do faktycznej wartości atrybutu. Niepewność ta może być modelowana z użyciem rozkładów możliwości; por. 2.1.2.

Formalnie, przypisanie wartości atrybutowi może być utożsamione z zadehowskim wyrażeniem lingwistycznym "X jest F" (por. (2.119)), gdzie X jest zmienną lingwistyczną (por. p. 2.3) odpowiadającą atrybutowi, zaś F jest (dysjunktywnym[7]) zbiorem rozmytym reprezentującym niedoskonałą informację o wartości tego atrybutu. Wtedy kombinacje różnych form niedoskonałej informacji mogą być reprezentowane z użyciem odpowiednich postaci wyrażeń lingwistycznych z kwalifikatorami (por. s. 45). Wyrażenia te mogą być przekształcone (por. s. 45) na wyrażenia w formie "X jest B", przy czym postać zbioru rozmytego B jest funkcją zbioru F i kwalifikatora. Tak więc, podstawowa forma wyrażenia

[7] por. s. 26

lingwistycznego "X jest F" odgrywa fundamentalną rolę w reprezentacji informacji niedoskonałej.

Model posybilistyczny

W podejściu posybilistycznym dysjunktywny zbiór rozmyty reprezentuje *nieprecyzyjnie* określoną wartość atrybutu A. Zbiór ten interpretowany jest jako rozkład możliwości π_A określony na dziedzinie atrybutu A, dom_A. Stopień możliwości, że dany element x dziedziny dom_A, $x \in dom_A$, jest faktyczną wartością atrybutu A jest równy $\pi_A(x)$. Pomysł zastosowania rozkładów możliwości do reprezentacji wartości atrybutów pojawia się już w pracy Umano [214], ale najbardziej kompletne podejście, najczęściej cytowane w literaturze, zaproponowali Prade i Testemale [182, 183, 184].

Potrzeba reprezentacji w bazie danych takiej nieprecyzyjnej informacji pojawia się zazwyczaj wtedy, gdy wartość atrybutu wyrażona jest z użyciem terminu lingwistycznego. Na przykład, załóżmy że data powstania obrazu nie jest dokładnie znana, ale wiadomo że powstał on "na początku czternastego wieku". Taka informacja może być dogodnie reprezentowana z użyciem następującego przykładowego rozkładu możliwości:

$$\pi_{\texttt{data}}(x) = \mu_{początek\_XIV\_wieku}(x) = \begin{cases} 0 & \text{jeśli } x \leq 1280 \\ \dfrac{x - 1280}{20} & \text{jeśli } 1280 < x \leq 1300 \\ 1 & \text{jeśli } 1300 < x \leq 1320 \\ \dfrac{1340 - x}{20} & \text{jeśli } 1320 < x \leq 1340 \\ 0 & \text{jeśli } x > 1340 \end{cases}$$

gdzie $\mu_{początek\_XIV\_wieku}$ oznacza funkcję przynależności zbioru rozmytego reprezentującego znaczenie terminu "początek czternastego wieku".

Użycie rozkładów możliwości jako wartości atrybutów stanowi oczywiste uogólnienie pseudowartości `null`, jak i jej wariantów takich jak *lub-zbiory*. W szczególności, pseudowartość $\texttt{null}_{\exists}$ jako wartość atrybutu A można reprezentować z użyciem następującego rozkładu możliwości:

$$\pi_A(x) = 1, \qquad \forall\ x \in dom_A$$

Dla *lub-zbioru* z przykładu (3.19) odpowiednikiem jest następujący rozkład możliwości:

$$\pi_{\texttt{adres}}(x) = \begin{cases} 1 & \text{jeśli } x \in \{\texttt{Warszawa}, \texttt{Poznań}, \texttt{Wrocław}\} \\ 0 & wpp \end{cases}$$

Reprezentacja pseudowartości $\texttt{null}_{\perp}$ wymaga rozszerzenia dziedziny dom_A każdego z atrybutów A o dodatkowy element $\perp_A$; por. np. [182, 77]. Można wtedy stosować trzy rozkłady posybilistyczne π_A^{UNK}, $\pi_A^{N/A}$ and π_A^{UNA} o następującej semantyce:

1. Wartość A jest nieznana, ale istnieje ($\texttt{null}_{\exists}$):
$$\pi_A^{UNK}(x) = \begin{cases} 1, \text{ dla } x \in dom_A \setminus \{\perp_A\} \\ 0, \text{ dla } x = \perp_A \end{cases}$$

2. Wartość A nie istnieje ($\texttt{null}_{\exists}$):
$$\pi_A^{N/A}(x) = \begin{cases} 0, \text{ dla } x \in dom_A \setminus \{\perp_A\} \\ 1, \text{ dla } x = \perp_A \end{cases}$$

3. Całkowity brak informacji co do wartości A:
$$\pi_A^{UNA}(x) = 1, \quad \forall x \in dom_A.$$

Reprezentacja danych w modelu posybilistycznym jest podobna do podejścia z zastosowaniem *lub-zbiorów*, ale pozwala na wyrażenie dla każdego elementu dziedziny atrybutu dodatkowej informacji co do stopnia możliwości, że jest on faktyczną wartością tego atrybutu dla danej krotki.

Wyszukiwanie informacji w posybilistycznej bazie danych odbywa się z użyciem języka zapytań stosownie rozszerzonej algebry relacji. Nazwiemy ja *algebrą relacji posybilistycznych*. Na przykład operacja wyboru σ zaadoptowana jest następująco (por. klasyczną definicję (3.6)). Rozważa się dwie formy warunków prostych:

(i) $A\ \theta\ a$, gdzie A jest atrybutem, θ jest operatorem porównania (klasycznym lub rozmytym, takim jak "$\leq$" czy "dużo większe"), zaś a jest stałą (skalarem z dziedziny atrybutu, $a \in dom_A$, lub zbiorem rozmytym zdefiniowanym na tej dziedzinie, $a \in \mathcal{F}(dom_A)$);

(ii) $A_i\ \theta\ A_j$, gdzie A_i i A_j są atrybutami.

W ogólnym przypadku, wartość atrybutu nie jest znana dokładnie (jest reprezentowana z użyciem rozkładu posybilistycznego), a więc spełnienie warunku jest wyrażone w terminach jego *możliwości* i *konieczności*, zgodnie z teorią możliwości; por. p. 2.1.2. Wzory (2.37) są stosowane do obliczenia stopni możliwości i konieczności w następujący sposób.

W przypadku (i) rozkład możliwości $\pi_{A(t)}(\cdot)$, reprezentujący wartość atrybutu A dla krotki t, stosowany jest do obliczenia miar możliwości i konieczności zbioru F elementów dziedziny dom_A, które są w relacji

θ z elementami reprezentującymi stałą a. Zbiór F jest w ogólności zbiorem rozmytym o następującej funkcji przynależności:

$$\mu_F(x) = \sup_{y \in dom_A} \min(\mu_\theta(x,y), \mu_a(y)), \qquad x \in dom_A$$

i zgodnie z (2.37):

$$\Pi_{A(t)}(F) = \sup_{x \in dom_A} \min(\pi_{A(t)}(x), \mu_F(x)) \tag{3.20}$$

$$\mathrm{N}_{A(t)}(F) = \inf_{x \in dom_A} \max(1 - \pi_{A(t)}(x), \mu_F(x)) \tag{3.21}$$

$$\tag{3.22}$$

gdzie $\mu_a(\cdot)$ oznacza funkcję przynależności zbioru rozmytego reprezentującego stałą[8] a, zaś $\mu_\theta(\cdot)$ reprezentuje rozmyty operator porównania (relację rozmytą) θ. Stopień przynależności krotki t do relacji będącej wynikiem zastosowania operacji wyboru jest wtedy parą $(\Pi_{A(t)}(F), N_{A(t)}(F))$.

W przypadku (ii) *łączny rozkład możliwości* $\pi_{(A_i(t),A_j(t))}$ (por. (2.43)) stosuje się do obliczenia miar możliwości i konieczności zbioru rozmytego F zdefiniowanego w przestrzeni iloczynu kartezjańskiego dziedzin atrybutów A_i i A_j, który obejmuje pary elementów tych dziedzin będące w relacji θ. Funkcja przynależności zbioru F ma następującą postać:

$$\mu_F(x,y) = \mu_\theta(x,y), \qquad x \in dom_{A_i}, \quad y \in dom_{A_j}$$

i zgodnie z (2.37):

$$\Pi_{(A_i(t),A_j(t))}(F) = \sup_{(x,y) \in A_i \times A_j} \min(\pi_{(A_i(t),A_j(t))}(x,y), \mu_F(x,y)) \tag{3.23}$$

$$\mathrm{N}_{(A_i(t),A_j(t))}(F) = \inf_{(x,y) \in A_i \times A_j} \max(1 - \pi_{(A_i(t),A_j(t))}(x,y), \mu_F(x,y)) \tag{3.24}$$

Stopień przynależności krotki t do relacji będącej wynikiem operacji wyboru ma wtedy postać pary $(\Pi_{(A_i(t),A_j(t))}(F), N_{(A_i(t),A_j(t))}(F))$. Jeśli przyjmiemy, że atrybuty A_i i A_j są nieinteraktywne (por. (2.47)), to obliczanie miar możliwości i konieczności (3.23)-(3.24) ulega uproszczeniu:

$$\pi_{(A_i(t),A_j(t))}(x,y) = \min(\pi_{A_i(t)}(x), \pi_{A_j(t)}(y))$$

[8] Jeśli stała a jest wielkością skalarną, elementem dziedziny dom_A, to reprezentujący ją zbiór rozmyty (oznaczmy go dla odróżnienia od stałej jako $\tilde{a}$) ma funkcję przynależności postaci $\mu_{\tilde{a}}(a) = 1$ i $\mu_{\tilde{a}}(x) = 0 \ \forall x \neq a$.

i stąd:

$$\Pi_{(A_i(t),A_j(t))}(F) = \sup_{(x,y)\in A_i\times A_j} \min(\pi_{A_i(t)}(x), \pi_{A_j(t)}(y), \mu_F(x,y)) \quad (3.25)$$

$$\mathrm{N}_{(A_i(t),A_j(t))}(F) = \inf_{(x,y)\in A_i\times A_j} \max(1-\pi_{A_i(t)}(x), 1-\pi_{A_j(t)}(y), \mu_F(x,y)) \quad (3.26)$$

Prade i Testemale [182, 183, 184] przyjmują, że wynikiem zapytania (sekwencji operacji algebry relacji posybilistycznych) są dwie (rozmyte) relacje:

- jedna obejmuje te krotki, które *na pewno* są wynikiem zapytania, przy czym stopień pewności wyraża wartość $N_{A(t)}(F)$;
- druga obejmuje te krotki, które *być może* są wynikiem zapytania, przy czym stopień możliwości wyraża wartość $\Pi_{A(t)}(F)$.

Należy zauważyć, że wynikiem operacji opisanej tu algebry relacji posybilistycznych (w tym operacji wyboru) są relacje rozmyte, przy czym stopnie przynależności krotek wyrażone są tradycyjnie pojedynczymi liczbami (jeśli interpretujemy wynik jako dwie relacje, jak to przedstawiono wyżej) lub parami liczb, wartościami miar możliwości i konieczności odpowiednich zbiorów rozmytych. W tym ostatnim przypadku należy określić pewien porządek w przestrzeni takich par liczb, tak żeby można było przedstawić użytkownikowi w pierwszej kolejności krotki najlepiej odpowiadające warunkom zapytania. Dyskutujemy szerzej analogiczny problem w przypadku zapytań bipolarnych w p. 5.2.

Prade i Testemale rozważają także operacje wyboru ze złożonym warunkiem C, np. postaci: $C = C_1 \wedge C_2$ lub $C = C_1 \vee C_2$ czy $C = \neg C_1$. Warto zwrócić uwagę na fakt, że w ogólności system wnioskowania dotyczącego niepewności, taki jak teoria możliwości, nie może być *ekstensjonalny* (ang. *truth-functional*) (por. np. [99]). W związku z tym, w ogólności – bez przyjęcia dodatkowych założeń, nie wystarczy obliczyć wartości miar możliwości i konieczności dla warunkow prostych, a następnie je zagregować z użyciem odpowiedniego operatora logicznego koniunkcji czy alternatywy. Takie postępowanie jest efektywne obliczeniowo, ale wymaga przyjęcia założenia o nieinteraktywności atrybutów występujących w warunku operatora wyboru.

Pozostałe operacje algebry relacji są również dostosowywane do wymagań modelu posybilistycznego. Niektóre z operacji nie wymagają żadnych zmian. Należy do nich na przykład iloczyn kartezjański. Stosując

złożenie wyżej opisanej adaptacji operatora wyboru z operacją iloczynu kartezjańskiego otrzymuje się natychmiast operację złączenia. Operacje teoriomnogościowe i operacja rzutu wymagają adaptacji pojęcia redundancji. Klasyczny warunek, że wartości wszystkich atrybutów redundantnych krotek są identyczne wymaga uszczegółowienia, gdyż wymaganie identyczności rozkładów możliwości jest w ogólnym przypadku niepraktyczne. Prade i Testemale wprowadzają pewną miarę podobieństwa dwóch rozkładów (zbiorów rozmytych), ale nie poświęcają temu zgadnieniu wiele miejsca. W literaturze zaproponowano w ostatnich latach wiele podejść do określania podobieństwa z użyciem aparatu logiki rozmytej, które mogą tu znaleźć zastosowanie. Jeden z ciekawszych przeglądów w tym zakresie stanowi monografia Cross i Sudkampa [72].

Podejście Prade'a i Testemale [182, 183, 184] jest najbardziej reprezentatywne dla prób modelowania w bazie danych informacji niedoskonałej z użyciem aparatu teorii możliwości. Inne podejścia przedstawione zostały między innymi w następujących pracach [214, 215, 262, 261].

Bosc i Pivert [41, 33] wprowadzili również inny typ zapytań względem posybilistycznej bazy danych. Pozwalają one wyszukać krotki, w których rozkłady możliwości będące wartościami wybranych atrybutów mają określone parametry. Tak więc, takie zapytania nie odnoszą się do wartości atrybutów jako takich, ale do charakterystyk związanych z nimi rozkładów możliwości. Użyteczność takich zapytań ilustrują następujące przykłady:

I. Znajdź krotki dla których wszystkie następujące wartości: $a_1, a_2, \ldots, a_n$ są możliwymi wartościami atrybutu A;
$\forall \ \ a_{i\in 1\ldots n}\pi_A(a_i) > 0.$

II. Znajdź krotki dla których przynajmniej n elementów dziedziny jest możliwą wartością atrybutu A w stopniu przynajmiej λ;
$|\{a : \pi_A(a) \geq \lambda\}| \geq n.$

III. Znajdź krotki dla których element a_1 jest bardziej możliwy jako wartość atrybutu A niż element a_2;
$\pi_A(a_1) > \pi_A(a_2).$

IV. Znajdź krotki dla których dla atrybutu A istnieje tylko jeden element całkowiecie możliwy jako jego wartość;
$|\{a : \pi_A(a) = 1\}| = 1.$

Stopnie spełnienia tego typu zapytań dla poszczególnych krotek obliczane są w dość oczywisty sposób. Na przykład dla zapytania typu I sto-

pień spełnienia zapytania będzie utożsamiony z najmniejszą spośród wartości $(\pi_A(a_1), \ldots, \pi_A(a_n))$.

Bosc i Pivert [42, 43] zaproponowali również zmiany w modelu posybilistycznym zmierzające do zapewnienia jego odpowiednich własności jako *systemu reprezentacji* (por. def. 3.2).

Model oparty na relacji podobieństwa.

Buckles i Petry [52] zaproponowali inne podejście z użyciem aparatu logiki rozmytej do modelowania informacji niedoskonałej w bazie danych. Punktem wyjścia jest obserwacja, że elementy należące do dziedziny danego atrybuty mogą być do siebie podobne i może się zdarzyć, że trudno ustalić, który z nich faktycznie odpowiada wartości atrybutu w danej krotce. W ogólności więc przyjmuje się, że wartości atrybutów są podzbiorami ich domen, a nie ich pojedynczymi elementami. Dodatkowo, dla dziedziny dom_A każdego atrybutu A zdefiniowana jest rozmyta relacja podobieństwa $S_A \in \mathcal{F}(dom_A \times dom_A)$, określająca stopień podobieństwa dla każdej pary elementów $x, y \in dom_A$:

$$\mu_{S_A} : dom_A \times dom_A \longrightarrow [0, 1]$$

Wartość $\mu_{S_A}(x, y) = 0$ oznacza, że elementy x i y są całkowicie różne, natomiast $\mu_{S_A}(x, y) = 1$ oznacza, że są one całkowicie podobne, nierozróżnialne. Buckles i Petry oryginalnie zakładali, że relacja rozmyta S_A jest rozmytą relacją równoważności, czyli jest *zwrotna*, *symetryczna* i *przechodnia*; por. def. 2.14. W kolejnych pracach stwierdzono jednak, że to założenie jest zbyt silne – ogranicza ono bardzo swobodę w określaniu podobieństwa elementów. Shenoi i Melton [201] zaproponowali rezygnację z wymogu przechodniości[9] zachowując przy tym wszystkie własności podejścia Bucklesa i Petry'ego. Zaadaptowano między innymi pojęcie redundancji krotek i operacje algebry relacji.

Podejście Bucklesa i Petry'ego bazuje na pojęciu *nierozróżnialności* elementów dziedziny danego atrybutu, które stanowi podstawę *teorii zbiorów przybliżonych* Pawlaka [176]. Stało się to podstawą do rozwinięcia tego podejścia do postaci modelu opartego na teorii Pawlaka [7, 8].

Model posybilistyczny i model oparty na relacji podobieństwa stanowią klasyczne już modele danych opracowane z użyciem aparatu pojęciowego logiki rozmytej. W literaturze zaproponowane również modele mieszane. Na przykład Medina, Pons i Vila [164] zaproponowali rozmyty

[9] Co daje w efekcie relację *bliskości* (ang. *proximity relation, tolerance relation*).

model danych GEFRED (ang. *Generalized Fuzzy Relational Database*), w którym dane reprezentują relacje rozmyte, wartościami atrybutów są zbiory rozmyte/rozkłady możliwosci i, dodatkowo, z każdą wartością atrybutu może być skojarzony stopień zgodności.

ISSN 0208-8029
ISBN 83-894-7551-0

INSTYTUT BADAŃ SYSTEMOWYCH
POLSKIEJ AKADEMII NAUK
tel.: (+48) 22 3810246 / 22 3810277 / 22 3810241 / 22 3810273
e-mail: biblioteka@ibspan.waw.pl