STUDIA INFORMATICA 2010
Volume 31 Number 2A (89)

Krzysztof GOCZYŁA, Aleksander WALOSZEK,
Wojciech WALOSZEK, Teresa ZAWADZKA
Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki

WNIOSKOWANIE Z RÓŻNYCH ŹRÓDEŁ OSOBNIKÓW W SYSTEMIE RKASEA

Streszczenie. Niniejszy artykuł prezentuje koncepcję zarządzania wiedzą asercjonalną zastosowaną w systemie zarządzania wiedzą RKASEA. Koncepcja ta traktuje opis świata jako zbiór źródeł osobników. Dzięki temu udało się objąć nią również mechanizm obsługi reguł oraz mechanizm pozyskiwania wiedzy z zewnętrznych źródeł danych. Mechanizmy te są traktowane jako dodatkowe typy źródeł osobników.

Słowa kluczowe: bazy wiedzy, *Semantic Web*, system wnioskujący, wiedza, reguły, logika opisowa

REASONING FROM MULTI-TYPED SOURCES OF INDIVIDUALS IN RKASEA

Summary. The paper presents the concept of managing assertional knowledge in a novel knowledge management system RKASEA. According to this concept the description of the world is handled as a set of sources of individuals. This concept turned out to be flexible enough to cover the mechanism for inferring with rules and the mechanism for acquiring the data from external data sources. In the presented framework the both mechanisms are recognized as additional types of sources of individuals.

Keywords: knowledge bases, *Semantic Web*, inference system, knowledge, rules, description logics

1. Wprowadzenie

System RKASEA jest rozwinięciem systemu KASEA [2]. Jego celem było:

- zwiększenie ekspresywności obsługiwanego języka logiki opisowej,

- odejście od założenia monotoniczności ontologii,
- dodanie obsługi wiedzy wyrażonej za pomocą reguł,
- rozwinięcie metod obsługi ontologii modularnych,
- dodanie obsługi rozproszenia wiedzy, w tym zewnętrznych źródeł danych nieontologicznych (ZŹD),
- umożliwienie uwzględnienia różnego poziomu wiarygodności opisu faktów

Aby sprostać wysokim wymaganiom wynikającym z tych celów, opracowane zostały nowe metody zarządzania i reprezentowania wiedzy. Zaproponowano konglomeratowy model baz wiedzy jako ogólną podstawę teoretyczną oraz praktyczną reprezentację takiej bazy w postaci drzew konglomeratów. W niniejszym opracowaniu przedstawiony zostanie przyjęty sposób obsługi wiedzy opisującej fakty. Właśnie ta część wiedzy nastręczała szczególnie wiele problemów realizacyjnych. Zderzało się tu bowiem wiele sprzecznych ze sobą wymagań. Można wyodrębnić trzy główne grupy problemów:

1. Pogodzenie obsługi opisu faktów w postaci części asercjonalnej ontologii wyrażonej w logice opisowej z opisem wyrażonym w postaci reguł.
2. Konieczność modyfikowania wniosków wyciągniętych wcześniej z powodu odejścia od założenia monotoniczności bazy wiedzy.
3. Uwzględnienie w procesie wnioskowania obsługi ZŹD. Połączenie z takim źródłem uniemożliwia zachowanie zasady monotoniczności oraz utrudnia wnioskowanie w przód.

W punkcie 2 opracowania opisane zostaną zagadnienia wstępne dotyczące logiki opisowej, baz ontologiczno-regułowych oraz integrowania danych z zewnętrznych źródeł. Punkt 3 dostarczy informacji o modelu konglomeratowym i algebrze z nim związanej oraz o języku KQL, bazującym na algebrze konglomeratów. W punkcie 4 zostaną opisane rozwiązania zastosowane w systemie RKASEA. Punkt 5 porównuje te rozwiązania z innymi zastosowanymi w znanych systemach wnioskujących, a punkt 6 podsumowuje opracowanie.

2. Zagadnienia wstępne

2.1. Logika opisowa

Logika opisowa (DL — od ang. *Description Logics*) [3] jest dziedziną nauki zajmującą się rozstrzygalnymi podzbiorami klasycznego rachunku predykatów (podzbiory te nazywamy *dialektami* DL). Logika opisowa jest w ramach inicjatywy *Semantic Web* [1] wykorzystywana do definiowania baz wiedzy zwanych *ontologiami*. Ontologia składa się z dwóch podzbiorów zdań. Jeden z podzbiorów zawiera zdania definiujące pojęcia ogólne zwane konceptami i rolami. Zdania te, zwane aksjomatami, mają formę opisu subsumpcji, np. $C \sqsubseteq D$, co oznacza, że koncept C dziedziczy od konceptu D. Koncept atomowy to po prostu nazwa,

koncept złożony jest określany złożonym wyrażeniem, np.: $\neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C$. Część ta nazywana jest terminologią lub T-boxem. Drugi z podzbiorów zawiera zdania zwane asercjami. Zdania te opisują fakty i mają postać $C(a)$ lub $R(a, b)$. Pierwsze ze zdań oznajmia, że obiekt a (zwany w ontologii *osobnikiem*) jest wystąpieniem konceptu C, drugie, że para osobników (a, b) jest wystąpieniem roli R. Ta część ontologii nazywana jest częścią asercjonalną, opisem świata, opisem faktów lub, symbolicznie, A-boxem.

Wspólną cechą dialektów logiki opisowej jest fakt, że operują one pojęciem modelu przypisującego terminom (konceptom i rolom oraz osobnikom) ich interpretacje (konceptom podzbiory dziedziny Δ, rolom podzbiory $\Delta \times \Delta$, osobnikom elementy Δ). Zdania formułowane w danym dialekcie stanowią zbiór ograniczeń, które ograniczają przestrzeń możliwych interpretacji. Na przykład przecięcie konceptów $C \sqcap D$ jest interpretowane następująco: $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$. W ogólności zbiór możliwych interpretacji (zakresów) danego konceptu (i roli) jest nieskończony, a o zależnościach pomiędzy terminami (np. podrzędności lub rozłączności konceptów, czy spełnialności pojedynczego konceptu) wnioskujemy, analizując zależności zachodzące między ich zakresami *we wszystkich* dopuszczalnych modelach. Specjalną rolę pełnią koncepty wyliczeniowe i atrybuty. Koncepty wyliczeniowe mają postać $\{a_1, a_2, \ldots a_n\}$ (gdzie a_i, $i \in [1 \ldots n]$, to nazwy osobnków), czyli są zbiorami osobników o znanej zawartości. Atrybuty są specjalnym typem roli, która wiąże osobniki dziedziny Δ z wartościami jakiejś dziedziny konkretnej $\Delta^{\mathcal{D}}$. Dziedziną konkretną może być zbiór liczb naturalnych lub rzeczywistych, zbiór łańcuchów tekstowych itp.

2.2. Rozszerzenie logiki opisowej o reguły

Alternatywną w stosunku do logiki opisowej i cieszącą się znacznie dłuższą historią metodą reprezentacji są tzw. klauzule Horna. Klauzule Horna (zwane dalej regułami) również stanowią inny niż DL podzbiór logiki pierwszego rzędu. W istocie reguły i DL mają część wspólną, a więc niektóre stwierdzenia logiczne można wyrazić zarówno w DL, jak i odpowiednią regułą.

Jednak istnieje spory obszar semantyczny, który jest rozłączny. W klasycznych (rozstrzygalnych) dialektach DL utrudnione jest wyrażanie takich reguł, w których liczba zmiennych jest większa niż jeden, co prowadzi do znacznego zubożenia możliwości ontologicznego reprezentowania rzeczywistości.

Te problemy zostały dostrzeżone również w ramach prac nad inicjatywą Sieci Semantycznej (*Semantic Web*). Najpierw opracowano język definiowania reguł RuleML o ekspresji klauzul Horna rozszerzonej o elementy proceduralności, a następnie zaproponowano język SWRL (Semantic Web Rule Language), stanowiący swoiste połączenie OWL i RuleML.

Język SWRL integruje ze sobą konstrukcje OWL i RuleML, nadając im jednolitą interpretację semantyczną. Niestety, ekspresywność SWRL-a jest na tyle duża, że jest on nierozstrzygalny. Z tego względu w praktyce stosuje się uproszczenia, ograniczając ekspresywność języka poprzez zdefiniowanie dodatkowych zasad semantycznej interpretacji reguł (czego przykładem jest tzw. zasada bezpiecznych reguł DL, polegająca na dodatkowym założeniu, że reguły dotyczą wyłącznie osobników, których nazwy zostały jawnie wymienione przez użytkownika) bądź też rezygnuje się z semantycznej jednolitości na rzecz podejścia hybrydowego [4] i do obsługi reguł stosuje się odrębne zasady i narzędzia.

2.3. Obsługa danych z zewnętrznych źródeł danych

Jednym z podstawowych problemów rozwijającej się Sieci Semantycznej jest brak wystarczającej liczby źródeł semantycznych. Dlatego jeden z nurtów prac zajmuje się integrowaniem wiedzy z takich źródeł do baz wiedzy. W dziedzinie przetwarzania źródeł zapisanych w języku naturalnym są również prowadzone intensywne prace, w ramach których na szczególną uwagę zasługują te związane z adnotowaniem tekstu [5] i przetwarzaniem tekstu w celu automatycznego budowania sieci semantycznych [6], [7]. W dziedzinie „podłączania" danych ze źródeł strukturalnych trwają intensywne prace, których celem jest odwzorowywanie baz danych na ontologie [8], [9]. Rozwiązanie zaproponowane w systemie RKASEA wpisuje się w drugi z wymienionych nurtów i jest oparte na opisanym w [10] rozwiązaniu polegającym na budowaniu warstwy wiedzy (ang. *Knowledge Layer*). Rozwiązanie to zostało wybrane ze względu na jego dwie podstawowe cechy:

- niezależność od typu (pół)strukturalnego źródła danych,
- jednolite postrzeganie przez system dowolnego źródła.

Szczególną rolę odgrywa pierwsza z wymienionych cech, pozwalająca na tworzenie dowolnych (pół)strukturalnych źródeł wiedzy i podłączanie ich do systemu wiedzy. Rozwiązanie to uzyskano w wyniku tworzenia odwzorowań pomiędzy terminami ontologicznymi a zbiorami danych zapisanymi w zewnętrznych źródłach. Podstawą wykorzystanego rozwiązania jest metoda SED (ang. *Semantic Enrichment of Data*). Sposób uzyskiwania tych źródeł danych nie wpływa na działanie algorytmów wnioskujących, gdyż dane są postrzegane zawsze jako zbiory, dzięki czemu uzyskuje się drugą z wymienionych cech.

3. Algebra konglomeratów i język KQL

Algebra konglomeratów jest formalizmem dla ostatnio wprowadzonej koncepcji modularyzacji baz wiedzy [11]. Rolę podstawowej jednostki pełni tutaj konglomerat. Konglomerat

definiowany jest ściśle semantycznie jako klasa dopuszczalnych modeli pewnego zbioru terminów (*słownika konglomeratu*). Taka definicja pozwoliła na zdefiniowanie działań algebraicznych w przestrzeni konglomeratów bardzo przypominających te wykorzystane w algebrze relacyjnej Codda [12]. Za pomocą wprowadzonego zestawu operacji możliwe jest bardzo elastyczne manipulowanie poszczególnymi jednostkami wiedzy i składanie z nich nowych jednostek dostosowanych do potrzeb użytkownika.

Użytkownik nie posługuje się wprost algebrą konglomeratów, lecz utworzonym na jej bazie językiem KQL. Język ten znacznie różni się od innych języków używanych do formułowania zapytań ontologicznych. Jest w swojej idei podobny do języka SQL. Daje możliwość formułowania rozkazów definiujących składniki bazy wiedzy, takich jak: schemat, konglomerat, reguła, odwzorowanie dla zewnętrznego źródła danych. Pozwala też formułować zapytania, które są domknięte ze względu na zbiór konglomeratów. Zapytanie sformułowane w KQL-u czerpie dane z istniejących w bazie wiedzy konglomeratów i zwraca nowy konglomerat. Szerszy opis języka KQL dostępny jest w publikacji [13].

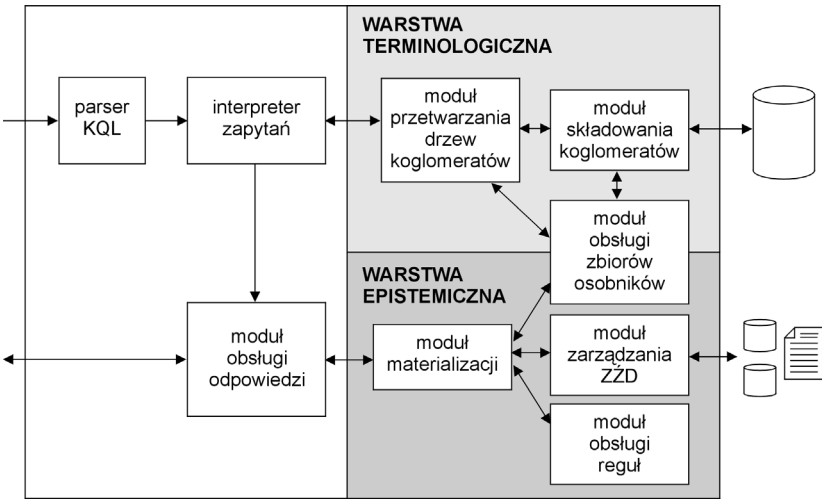
4. Zarządzanie faktami w systemie RKASEA

4.1. Architektura systemu RKASEA

Przyjęte w systemie RKASEA rozwiązania nakierowane są na realizację konglomeratowej koncepcji bazy wiedzy. Jednakże należy zwrócić uwagę na to, że większa część konglomeratów to twory dynamiczne, zmieniające się w czasie. W każdej chwili użytkownik ma prawo zmodyfikować zarówno słownik, jak i zbiór interpretacji dla danego konglomeratu. Zmiana może być spowodowana dodaniem aksjomatu, który dokłada nowe koncepty lub role i/lub ustanawia nowe zależności między termami. Mogą być dodawane asercje zmieniające zawartość A-boxa. Mogą być ładowane do bazy wiedzy nowe dane z zewnętrznych źródeł danych. W końcu reguły mogą propagować zmiany między konglomeratami, reagując na zmiany w jednym konglomeracie produkcją nowych asercji w innym.

Architektura systemu jest schematycznie przedstawiona na rys. 1. Na rysunku tym wyodrębniono moduły komunikacyjne. Wśród nich *parser KQL* odpowiada za niskopoziomową analizę zapytań KQL kierowanych do systemu, i*nterpreter zapytań* analizuje semantyczną treść zapytania i koordynuje wykonanie jego poszczególnych części. M*oduł obsługi odpowiedzi* współpracuje z użytkownikiem/klientem systemu w procesie udzielania odpowiedzi na zapytania. Sama baza składa się z dwóch warstw: *terminologicznej* i *epistemicznej*. Warstwa terminologiczna zawiera *moduł przetwarzania drzew konglomeratów* i *moduł składowania konglomeratów*. Obsługuje ona struktury reprezentujące definicje konglomeratów, a także definicje reguł i odwzorowań ZŹD. Z kolei warstwa epistemiczna zawiera

moduły *obsługi reguł*, *zarządzania zewnętrznymi źródłami danych* oraz *moduł materializacji konglomeratów*. Jest ona odpowiedzialna za wszelkie operacje związane z przechowywaniem faktów. Strzałki przedstawiają przepływy danych pomiędzy poszczególnymi częściami systemu. Bardziej szczegółowy opis elementów warstw terminologicznej i epistemicznej znajduje się w kolejnych punktach opracowania 4.2 i 4.3.



Rys. 1. Struktura systemu RKASEA
Fig. 1. The structure of RKASEA system

4.2. Warstwa terminologiczna

Warstwa terminologiczna jest pojęciem, które obejmuje szereg algorytmów i struktur danych odpowiedzialnych za obsługę wewnętrznej reprezentacji konglomeratów. Zagadnieniu reprezentowania konglomeratów jest poświęcone osobne opracowanie [14]. W tym miejscu można przyjąć, że reprezentacja konglomeratu jest, mówiąc w uproszczeniu, zbiorem grafów skierowanych, z których każdy opisuje jedną z dziedzin pomocniczych konglomeratu i jej podział na zakresy predykatów ze słownika konglomeratu tyczących się tej dziedziny. Dziedziną *główną* jest, zgodnie z założeniami logiki opisowej (patrz punkt 2.1), pewien zbiór obiektów Δ. Graf opisujący dziedzinę główną nazywamy *grafem* lub *drzewem konceptów*. Dziedziny *pomocnicze* to przede wszystkim dziedziny wartości atrybutów (dziedziny konkretne); każdy typ atrybutu wyznacza odrębną dziedzinę pomocniczą, która jest reprezentowana odrębnym grafem. Dodatkowo można wyróżnić grafy reprezentujące dziedziny par obiektów, przechowujące informację o zakresach predykatów binarnych odpowiadających rolom (*graf* lub *drzewo ról*) oraz grafy reprezentujące zakresy predykatów binarnych odpowiadających atrybutom. Zakresy tych ostatnich to zbiory par obiekt-wartość. Węzły każdego z grafów reprezentują podzbiory danej dziedziny w sensie teorii mnogości, odpowiadające zakresom poszczególnych terminów ze słownika konglomeratu, będących predykatami odpowiednimi dla danej dziedziny (głównej lub pomocniczej). Krawędzie

danego grafu reprezentują w tym ujęciu zależność zawierania się tych podzbiorów w sobie. Dodatkowo wszelkie nietaksonomiczne zależności między terminami wyrażone są za pomocą metody kartograficznej [15], co uzasadnia, dlaczego reprezentacja nazywa się *grafowo-kartograficzną*.

Funkcjonowanie grafów reprezentujących dziedziny konceptów i grafów reprezentujących dziedziny ról jest bardzo podobne. Dlatego też w dalszej części opracowania termin *osobnik* będzie używany zarówno w odniesieniu do wystąpienia konceptu, jak i w odniesieniu do wystąpienia roli lub atrybutu.

Z punktu widzenia tematu tego opracowania istotne jest opisanie sposobu zarządzania wiedzą o osobnikach. W warstwie terminologicznej zajmuje się tym przedstawiony na rys. 1 *moduł przetwarzania drzew konglomeratów* (*eMPeDoKles*). Wiedza o osobnikach dostarczana jest do systemu w postaci asercji (faktów). Z punktu widzenia algebry konglomeratów powodują one rozszerzenie słownika konglomeratu (poprzez dostarczenie nowych nazw osobników) oraz dokonują selekcji dopuszczalnych interpretacji (poprzez wyeliminowanie tych interpretacji, które dopuszczają istnienie faktów sprzecznych z dostarczonymi asercjami). Istnieją trzy źródła asercji: (i) użytkownik dostarczający asercji za pomocą rozkazów języka KQL, (ii) reguły oraz (iii) zewnętrzne źródła danych. Aby umożliwić jednolity sposób zarządzania tą wiedzą, wprowadzono pojęcie *źródła osobników*. Źródłem osobników nazywamy obiekt systemu, który jest odpowiedzialny za dostarczenie zbioru znanych systemowi osobników danego typu. Każdy węzeł w grafach konglomeratu dysponuje własnym źródłem osobników. Źródło może być *proste* lub *złożone*. Źródło proste to zwykły zbiór osobników, jedna reguła, jedno odwzorowanie ZŹD. Źródło złożone jest tworzone poprzez wykonanie operacji teoriomnogościowej przecięcia lub sumy na *n* źródłach prostych.

Źródło będące zwykłym zbiorem osobników, nazwane *źródłem wyliczeniowym*, powstaje, gdy do systemu poprzez parser KQL i interpreter zapytań dostarczane są asercje (zdania $C(a)$ lub $R(a, b)$ – patrz punkt 2.1). Najpierw w grafach konglomeratu odszukiwany jest węzeł odpowiadający użytemu w asercji konceptowi lub roli (C lub R). Jeśli taki węzeł nie istnieje, to jest tworzony i wstawiany we właściwe miejsce w reprezentacji konglomeratu. Argumenty predykatu, czyli osobnik (a) lub para osobników (a, b) trafiają do źródła osobników przypisanego temu węzłowi.

Źródło proste powiązane z regułą, czyli *źródło regułowe*, powstaje, w każdej sytuacji, gdy do bazy wiedzy zostaje wprowadzona reguła. Może to być wtedy, gdy użytkownik tworzy nową regułę, korzystając z rozkazu `CREATE RULE`. Duża ilość reguł powstaje też automatycznie, aby reprezentować pewne zależności, których nie jest w stanie objąć reprezentacja grafowo-kartograficzna. Przykładem takich reguł są reguły opisujące role przechodnie lub niektóre zależności między grafem ról i grafem konceptów (m.in. pokrywające zakres normalizacji stosowanej w metodzie kartograficznej, patrz [15]). Powstałe źródło

zostaje podłączone do węzła odpowiadającego nazwie predykatu tworzącego głowę reguły[1]. Produkcja wynikająca z tworzonej reguły zasila powstałe w ten sposób źródło w osobniki, które są argumentami predykatów w produkowanych faktach.

Źródło proste powiązane z odwzorowaniem ZŹD, nazwane *źródłem mapowanym*, powstaje, gdy użytkownik tworzy pojedyncze odwzorowanie, korzystając z klauzuli `ADD MAPPING` rozkazu `CREATE MAPPINGS`. Każde odwzorowanie przyporządkowuje fragment zewnętrznego zbioru danych terminowi ontologicznemu (patrz punkt 2.3), czyli pewnemu węzłowi grafów konglomeratu. Aktywacja odwzorowania produkuje osobniki o typie odpowiadającym temu węzłowi.

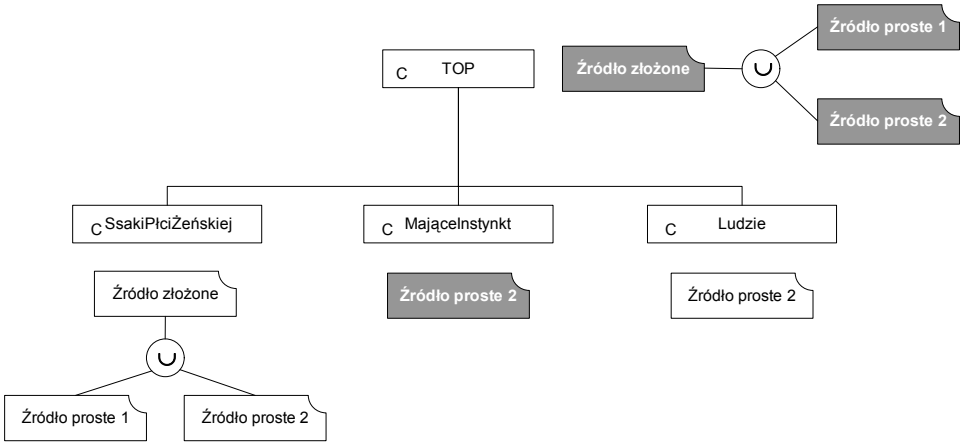
Zarządzanie osobnikami w warstwie terminologicznej sprowadza się do właściwego przypisania źródeł osobników do węzłów grafów konglomeratów. Nie jest tu istotna, poza jednym wyjątkiem, sama zawartość źródeł. Tym wyjątkiem są koncepty wyliczeniowe. Koncepty wyliczeniowe są pewnym typem źródła wyliczeniowego, którego zawartość jest znana w warstwie terminologicznej. W pozostałych przypadkach zakładamy, że na tym etapie jest ona nieznana.

W praktyce większość węzłów ma przypisane złożone źródła osobników. Źródła złożone powstają zawsze wtedy, gdy do reprezentacji jakiegoś konglomeratu dodawany jest nowy węzeł. Mówiąc w uproszczeniu, ma to miejsce wtedy, gdy użytkownik wysyła do systemu aksjomat lub asercję zawierającą nowy koncept (atomowy lub złożony) lub rolę (atomową lub złożoną) lub gdy w wyniku zapytania sformułowanego w języku KQL powstaje nowy konglomerat. W celu przypisania właściwych źródeł osobników do poszczególnych węzłów w drzewie zostały wykorzystane dwa mechanizmy: mechanizm przenoszenia „w górę” oraz mechanizm maksymalnego pokrycia, którego nazwa wywodzi się od nazwy wykorzystanego w tym mechanizmie algorytmu MCA (ang. *Maximum Coverage Algorithm*) [16]. Celem mechanizmu przenoszenia „w górę” jest przypisanie odpowiednich źródeł osobników zgodnie ze ścieżką dziedziczenia w grafie. Celem algorytmu MCA jest wyznaczenie dla danego konceptu C wszystkich takich przecięć konceptów, które dają w wyniku koncept dziedziczący od konceptu C. Aby to zobrazować posłużmy się przykładem. Niech zbiór uniwersum stanowią wszystkie ssaki. Wśród ssaków wyróżniamy te *MającеInstynkt*, *Ludzi* i *SsakiPłciŻeńskiej*. Ponadto do bazy wiedzy został dodany aksjomat definiujący powszechnie znany fakt, że wszystkie kobiety i niektórzy mężczyźni posiadają instynkt ($SsakiPłciŻeńskiej \sqcap Ludzie \sqsubseteq MającеInstynkt$). Zostało to zobrazowane na rysunku 2. Ponadto na rysunku pokazano przypisanie źródeł osobników do poszczególnych konceptów w drzewie. Źródła osobników przypisane jawnie do poszczególnych węzłów są oznaczone kolorami jasnymi, zmiany po zastosowaniu mechanizmów przenoszenia „w górę” i maksymalnego pokrycia zostały oznaczone kolorami ciemnymi. Przed zastosowaniem opisywanych mecha-

[1] Język KQL ogranicza liczbę atomów w głowie reguły do jednego.

nizmów do konceptu *SsakiPłciŻeńskiej* są przypisane dwa źródła osobników (1 i 2). Do konceptu *Ludzie* jest przypisane tylko jedno źródło (2). Mechanizm przenoszenia „w górę" wymaga przepisania wszystkich źródeł osobników do konceptów nadrzędnych. W naszym przykładzie wszystkie koncepty dziedziczą tylko od konceptu Top, dlatego to właśnie do tego konceptu należy przypisać oba źródła osobników.

Algorytm MCA ma inne zadanie, polegające na wyszukiwaniu właściwych źródeł, które nie wynikają bezpośrednio ze ścieżki dziedziczenia. Tutaj przykładem jest koncept *MającеInstynkt*, do którego mechanizm maksymalnego pokrycia przypisuje źródło osobników 2, które jest przypisane zarówno do konceptu *SsakiPłciŻeńskiej*, jak i konceptu *Ludzie*. To przypisanie wynika bezpośrednio z dodanego do bazy wiedzy aksjomatu. W rzeczywistych bazach wiedzy może istnieć wiele takich wywiedzionych przypisań, które choć wynikają z aksjomatów, niekoniecznie muszą być w nich bezpośrednio zapisane, jak to jest w prezentowanym przykładzie.



Rys. 2. Przykładowe drzewo konceptów z przypisanymi źródłami osobników
Fig. 2. An exemplary concept tree with individual sources assigned

Algorytm MCA wyszuka zawsze wszystkie przecięcia konceptów, które w wyniku dają koncept dziedziczący od zadanego konceptu. W [17] zostało dowiedzione, że na podstawie sumy takich przecięć możliwe jest sformułowanie wyrażenia łączącego źródła proste[2] za pomocą teoriomnogościowych operacji przecięcia i sumy zbiorów, zwracającego wszystkie te spośród dostępnych w źródłach nazw osobników, które są wystąpieniami konceptu C. W wersji oryginalnej algorytm bazował na pojęciu sygnatury wywodzącym się z kartograficznej reprezentacji wiedzy. W systemie RKASEA reprezentacja została zmieniona na grafowo-kartograficzną [14]. Zmiana ta nie wpływa bezpośrednio na działanie algorytmu, gdyż algorytm wymaga od reprezentacji jedynie standardowych metod sprawdzania rozłączności i podrzędności.

[2] W [14] algorytm dotyczył tylko odwzorowań ZŹD. Dodanie innych typów źródeł osobników nie powoduje konieczności zmiany dowodu.

4.3. Warstwa epistemiczna

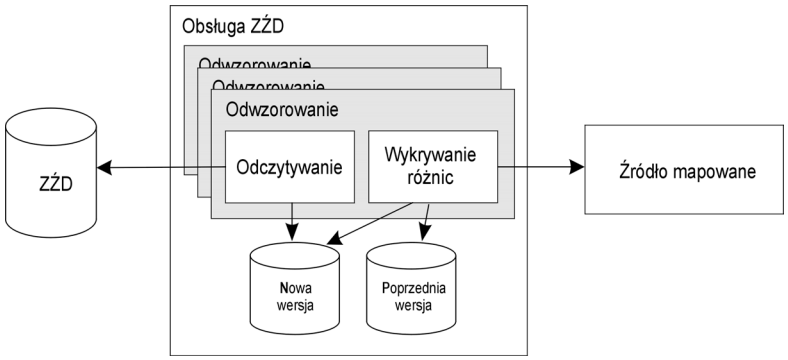
Warstwa epistemiczna zarządza samymi osobnikami. Jej główne zadania zdeterminowane są sposobem komunikacji z użytkownikiem, a konkretnie formą, w jakiej użytkownik otrzymuje odpowiedź na zapytanie. Jak wspomniano wyżej, zapytanie w języku KQL zwraca konglomerat. Konglomerat można traktować jak niezależną ontologię. Przyjętą dotychczas metodą korzystania z wiedzy zawartej w ontologii jest użycie silnika wnioskującego, który można odpytywać za pomocą jednego z ontologicznych języków zapytań, takich jak DIG, SPARQL czy OWL-QL. Takie rozwiązanie zostało uznane za niepraktyczne. Wymagałoby od użytkownika umiejętności posługiwania się dwoma językami i komplikowałoby korzystanie z bazy wiedzy w środowisku obiektowym. Należałoby bowiem najpierw uruchomić zapytanie KQL-owe, następnie zapytanie w języku zapytań ontologicznych, w końcu dokonać transformacji wyniku na obiekty. W systemie RKASEA użytkownik w odpowiedzi na zapytanie zadane w języku KQL otrzymuje odpowiedź w postaci obiektu `MaterializedConglomeration`, czyli *konglomeratu zmaterializowanego*. Jest to obiekt analogiczny do obiektu `ResultSet` w Javie odpowiedzialnego za dostarczenie użytkownikowi wyniku zapytania relacyjnego. Tak jak `ResultSet` nie jest relacją, tak `MaterializedConglomeration` nie jest konglomeratem, lecz dostarcza interfejsu umożliwiającego poznanie niektórych cech konglomeratu. Za jego pomocą można na przykład uzyskać listę nazw konceptów, ról lub atrybutów, listę osobników będących wystąpieniami danego konceptu atomowego, listę par osobników będących wystąpieniami danej roli atomowej, listę par obiekt-wartość będących wystąpieniami danego atrybutu. Obiekt `MaterializedConglomeration` jest więc pewnym domknięciem świata otwartego charakterystycznego dla ontologii. Jest też, w przeciwieństwie do konglomeratu, migawką (*snapshot*), posiada więc własną pamięć. Za jego utworzenie odpowiedzialny jest *moduł materializacji*.

W trakcie materializacji konglomeratu zawartość związanych z nim źródeł osobników jest kopiowana do pamięci obiektu `MaterializedConglomeration`. Taki sposób obsługi zapytań sprawia, że obowiązkiem warstwy epistemicznej jest przede wszystkim dbanie o to, aby proste źródła osobników dysponowały aktualnymi zbiorami osobników.

Jeśli chodzi o źródła wyliczeniowe, to są one wypełniane osobnikami już w warstwie terminologicznej. Warstwa terminologiczna odpowiada za umieszczenie takich osobników we właściwych węzłach grafu.

Źródłami regułowymi zarządza *moduł obsługi reguł*. Źródła te tworzą swoje zbiory osobników poprzez ich produkowanie w wyniku aktywacji reguły. Aby usprawnić ten proces, w module tym tworzona jest sieć Rete (ze względu na brak miejsca opisanie algorytmu Rete nie jest tu możliwe; Czytelnika odsyłamy np. do [18]). Nie jest ona związana z konkretnym konglomeratem, gdyż atomy reguł mogą korzystać z terminów zdefiniowanych w dowolnym konglomeracie. W opisywanym rozwiązaniu algorytm ten został tak zmodyfikowany, aby

węzły grafów konglomeratów mogły zostać użyte jako węzły sieci Rete, a źródła osobników jako pamięć faktów. Jeśli na przykład w źródle osobników związanym z rolą R znajduje się para osobników (a, b), to odpowiada to sytuacji, gdy w węźle o wzorze $R(?x, ?y)$ znajduje się fakt $R(a, b)$. Węzły grafów konglomeratów włączane do sieci Rete stają się częścią sieci alfa. W niektórych szczególnych przypadkach mogą też zostać włączone jako węzeł sieci beta. Może to mieć miejsce np. wtedy, gdy jeden z atomów reguły dotyczy jakiejś roli $R(?x, ?y)$, a drugi atom określa zakres dla drugiego argumentu tego predykatu, np. $C(?y)$. Wtedy, jeśli w grafie ról istnieje węzeł dla roli $R|C$ (tak określamy rolę, dla której dziedziną drugiego argumentu jest koncept C), to węzeł ten staje się węzłem łączącym w sieci beta. Węzły terminalne sieci Rete (węzły odpowiadające atomom z konkluzji reguł) są zawsze węzłami grafów konglomeratu. Sieć Rete jest aktywowana zawsze wtedy, gdy w którymś z węzłów sieci alfa znajdzie się nowy fakt; innymi słowy, gdy w źródle osobników związanym z węzłem grafu konglomeratu traktowanym jako węzeł sieci alfa zostanie umieszczony nowy osobnik. Aktywacja trwa do momentu, gdy w źródłach osobników przestaną pojawiać się nowe osobniki.



Rys. 3. Przykładowa obsługa zewnętrznego źródła danych
Fig. 3. An exemplary support for external data source

Źródła mapowane obsługuje *moduł zarządzania ZŹD*. Algorytm Rete wymaga zastosowania strategii wnioskowania w przód. Powoduje to konieczność wstępnego wczytania danych ze wszystkich zewnętrznych źródeł i późniejsze periodyczne aktualizowanie ich na zasadzie przyrostowej. Przyjęta koncepcja przewiduje, że obsługą pojedynczych zewnętrznych źródeł danych będą zajmowały się moduły uruchamiane w odrębnych wątkach. Wewnętrzna organizacja tych modułów może być różna, zależna od charakteru zewnętrznego źródła oraz od posiadanych w jego ramach uprawnień. W obecnej wersji systemu zrealizowana została obsługa ZŹD dla źródeł relacyjnych, funkcjonująca w sposób zilustrowany na rys. 3.

W ramach jednego źródła można utworzyć wiele odwzorowań. Każde odwzorowanie zaopatruje w osobniki inne źródło mapowane. Nowy odczyt osobników z zewnętrznego źródła jest wykonywany periodycznie, zgodnie z okresem zdefiniowanym dla tego źródła, i zapisywany w pamięci własnej modułu. Następnie nowa wersja zbioru osobników porówna-

wana jest z wersją poprzednią. Jeśli zostaną wykryte różnice między obiema wersjami, to zostaną one przekazane do obiektu reprezentującego źródło mapowane. Jeśli źródło to współpracuje z węzłem sieci Rete, to nastąpi uaktywnienie tej sieci i wykonane zostaną konieczne produkcje.

5. Porównanie z istniejącymi rozwiązaniami

Najbardziej znane silniki wnioskujące obsługujące SWRL to RacerPro [19], Pellet [20] i KAON2 [21]. Dwa pierwsze wykorzystują do wnioskowania algorytm *tableau* rozszerzony o obsługę reguł. Z kolei KAON2 zamienia ontologiczną część bazy wiedzy na zbiór reguł i w pełni korzysta z technik znanych dla dedukcyjnych baz wiedzy. W rozwiązaniu zaproponowanym w systemie RKASEA dokonano próby połączenia dwóch technik wnioskowania: ontologicznej i dedukcyjnej. Integracja obejmuje najniższy poziom struktur danych używanych w obu technikach. Dodatkowo uwzględnione zostały wymagania, jakie narzucają na system model konglomeratowy oraz obsługa zewnętrznych źródeł danych.

Główną różnicą między opisanym w tym opracowaniu rozwiązaniem a rozwiązaniami istniejącymi w zakresie baz wiedzy jest idea upodobnienia niektórych aspektów sposobu korzystania z nich do sposobu korzystania z relacyjnych baz danych.

6. Podsumowanie

W niniejszym opracowaniu przedstawione zostały rozwiązania, jakie w zakresie dotyczącym wnioskowania z asercjonalnej części bazy wiedzy zostały zaimplementowane w systemie RKASEA. Rozwiązania te integrują w sobie obsługę reguł oraz zewnętrznych źródeł osobników.

W systemie RKASEA w warstwie terminologicznej wykorzystano zaadaptowane elementy mechanizmów opracowanych na potrzeby warstwy wiedzy *KL*, przede wszystkim algorytm MCA. W warstwie epistemicznej w procesie materializacji konglomeratów wykorzystano adaptację znanego algorytmu Rete. Adaptacja ta polega na zintegrowaniu sieci alfa i beta używanych w tym algorytmie ze strukturami reprezentującymi konglomeraty.

Obecne rozwiązanie jest prototypowe i planowane jest jego rozwinięcie w stronę obsługi transakcyjnego przetwarzania wiedzy oraz wykorzystania mechanizmów alternatyw w celu pełniejszej obsługi konstrukcji algebry konglomeratów.

BIBLIOGRAFIA

1. Berners-Lee T., Hendler J., Lassila O.: The Semantic Web. Scientific American, nr 284(5), 2001, s. 34÷43.
2. Waloszek W.: Metody strukturalnej analizy ontologii opartych na logice opisowej. Praca doktorska, Gdańsk, 2008.
3. Baader F.A., McGuiness D.L., Nardi D., Patel-Schneider P.F. (red.): The Description Logic Handbook: Theory, implementation, and applications. Cambridge University Press, 2003.
4. Antoniou G. i in.: Combining Rules and Ontologies. A survey, raport projektu REWERSE dostępny pod adresem: http://rewerse.net/deliverables/m12/i3-d3.pdf.
5. Semantic MediaWiki: http://semanict-mediawiki.org
6. Vanderwende L., Kacmarcik G., Suzuki H, Meneses A.: MindNet: an automatically-created lexical resource, in HLT/EMNLP Interactive Demonstrations Proceedings, 2005, Redmond, WA 98052, USA.
7. ConceptNet: http://www.wolframalpha.com/
8. Barrasa J., Corcho Ó., Gómez-Pérez A.: R2O: An extensible and semantically based database-to-ontology mapping language, WebDB 2004. Proc. of the 7th Int. Workshop on the Web and Databases (2004).
9. Cullot N., Ghawi R., and Yetongnon K. : DB2OWL: A Tool for Automatic Database-to-Ontology Mapping. In Proceedings of 15th Italian Symposium on Advanced Database Systems (SEBD 2007), 17-20 2007, Torre Canne, Italy, s. 491÷494.
10. Goczyła K., Zawadzka T., Zawadzki M.: Managing Data from Heterogeneous data Sources using Knowledge Layer. Software Engineering Techiques: Design for Quality, IFIP International Federation for Information processing, Vol. 227, K. Sacha (red.), Boston, Springer, 2006, s. 300÷312.
11. Goczyła K., Waloszek A., Waloszek W.: Algebra konglomeratów jako narzędzie opisu problemów przetwarzania ontologii. Studia Informatica, Vol. 30, No. 2A, 2009, s. 141÷156.
12. Codd E.F.: Relational Completeness of Data Base Sublanguages. Database Systems, 1979, t. 6, s. 65÷98.
13. Goczyła K., Piotrowski P., Waloszek A., Waloszek W., Zawadzka T.: Język KQL jako realizacja idei języka SQL dla bazy wiedzy (praca zgłoszona do publikacji).
14. Goczyła K., Waloszek A., Waloszek W., Zawadzka T.: Wewnętrzna reprezentacja konglomeratowej bazy wiedzy w systemie RkaSeA (praca zgłoszona do publikacji).

15. Goczyła K., Waloszek W., Zawadzka T., Zawadzki M.: The Knowledge Cartography – a New Approach to Reasoning over Description Logics Ontologies. W: SOFSEM 2006: Theory and Practice of Computer Science, red. J. Wiedermann, J. Stuller, G. Tel, J. Pokorny, M. Bielikova, Berlin Heilderberg, Springer-Verlag, 2006 (LNCS 3831), s. 293÷302.
16. Goczyła K. Zawadzka T., Zawadzki M.: Wnioskowanie z danych zapisanych w zewnętrznych źródłach w systemie zarządzania wiedzą. Bazy Danych, Nowe technologie – Architektura, metody formalne i zaawansowana analiza danych: praca zbiorowa pod red. S. Kozielski, B. Małysiak, P. Kasprowski, D. Mrozek. Warszawa: WKŁ 2007. s. 283÷294.
17. Zawadzka T.: Integracja heterogenicznych źródeł wiedzy z wykorzystaniem logiki opisowej. Rozprawa doktorska, Politechnika Gdańska, 2009.
18. Forgy C.L.: Rete: A fast algorithm for the many pattern/many object pattern match problem, w: Artificial Intelligence, tom 19, nr 1, 1982, s. 17÷37.
19. Haarslev V., Möller R.: RACER System Description, w: Proceedings of IJCAR 2001, LNAI 2083, Springer-Verlag, 2001, s. 701÷706.
20. Sirin E., Parsia B.: Pellet: An OWL DL reasoner, w: Proceedings of the International Workshop on Description Logics, 2004, s. 212÷213.
21. KAON2, http://kaon2.semanticweb.org/

Recenzenci: Prof. dr hab. inż. Stanisław Kozielski
Prof. dr hab. inż. Andrzej Świerniak

Wpłynęło do Redakcji 31 stycznia 2010 r.

Abstract

The RKASEA system, created by KMG group from Gdansk University of Technology, meets the most sophisticated requirements posted before modern reasoners. On one hand it operates on knowledge bases expressed in SWRL language, i.e. lets to express knowledge using axioms or rules. On the other hand it allows to include information from external databases into the reasoning process. But such high requirements make the process very problematic as it engages different reasoning techniques and different ways of data acquisition. To solve these problems a new method of operating with individuals was developed. In this method sets of individuals are "produced" by individual sources. Three types of sources are established by (i) ontological statements about individuals, (ii) rules and

(iii) mappings to external databases. In the paper we show the method of integrating all the mechanisms on the level of used data structures and basic algorithms. The terminological layer of the system uses the adjusted MCA algorithm, introduced earlier for Knowledge Layer, to manage sources on the level of terminology representation. The epistemic layer materializes s-modules requested by a user and in this process it exploits a special adaptation of Rete pattern matching algorithm. In the adaptation alpha and beta networks are integrated with graphical-cartographic representation of the terminology.

Adresy

Krzysztof GOCZYŁA: Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki, Katedra Inżynierii Oprogramowania, ul. G. Narutowicza 11/12, 80-233 Gdańsk, Polska, kris@eti.pg.gda.pl.

Aleksander WALOSZEK: Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki, Katedra Inżynierii Oprogramowania, ul. G. Narutowicza 11/12, 80-233 Gdańsk, Polska, alwal@eti.pg.gda.pl.

Wojciech WALOSZEK: Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki, Katedra Inżynierii Oprogramowania, ul. G. Narutowicza 11/12, 80-233 Gdańsk, Polska, wowal@eti.pg.gda.pl.

Teresa ZAWADZKA: Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki, Katedra Inżynierii Oprogramowania, ul. G. Narutowicza 11/12, 80-233 Gdańsk, Polska, tegra@eti.pg.gda.pl.