Dotazovací jazyk SPARQL

(minimální základ + extrakt ze specifikací)

Doc. Ing. Vojtěch Svátek, Dr.

Zimní semestr 2017

http://nb.vse.cz/~svatek/rzzw.html

SPARQL

- **S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage ☺
- Standard W3C od 15. 1. 2008
  - SPARQL 1.1 od března 2013
  - http://www.w3.org/TR/sparql11-overview/ (rozcestník na podrobnější dokumenty)
- Dvě základní části (z hlediska 4IZ440)
  - Dotazovací jazyk pro RDF
  - Jazyk pro zápis informací (SPARUL – „SPARQL update language“), také standard z března 2013
    - http://www.w3.org/TR/sparql11-update/

Dotaz typu SELECT

- Základní typ dotazu, blízký SQL
- Obsahuje zejména
  - Seznam výstupních proměnných
    - Lze nahradit univerzální hodnotou (*)
  - Klauzuli FROM, identifikující zdrojová data
    - Datový graf lze ale deklarovat i ve WHERE
  - Klauzuli WHERE
    - Obsahuje seznam trojic tvořících tzv. „grafový vzor“ v jazyce Turtle (obohaceném o proměnné); lze explicitně určit graf
    - Může obsahovat další proměnné vedle těch z výstupního seznamu

Příklad dotazu (1)

```
SELECT ?title
WHERE {
        <http://example.org/book/book1>
        <http://purl.org/dc/elements/1.1/title>
        ?title .
        }
```

Další možnosti

- Deklarace prefixů (bez @)
- Vyjádření nepovinnosti části grafového vzoru (vnořená klauzule OPTIONAL)
- Filtrování nalezených výsledků dodatečnou podmínkou (vnořená klauzule FILTER)
- Spojení více vzorů disjunkcí (UNION)
- Řada dalších
  - Agregace (GROUP BY), uspořádání (ORDER BY), omezení počtu výsledků (LIMIT), regulární výrazy…

Příklad dotazu (2)

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?nameX ?nameY ?nickY
WHERE {
        ?x foaf:knows ?y ;   foaf:name ?nameX .
        ?y foaf:name ?nameY .
        OPTIONAL { ?y foaf:nick ?nickY . }
        FILTER ( lang(?nameY) = "es" )
        }
```

Jiné typy dotazů (mimo SELECT)

- ASK
  - Bez výstupních proměnných, vrací TRUE/FALSE
- CONSTRUCT
  - Vrací nový graf RDF
  - Transformační služba (a la XSLT pro XML…)
- DESCRIBE
  - Vrací „popis“ entity zadané na vstupu (graf jejího „okolí“ v RDF)
  - Tvar výsledku určuje služba poskytující data, nelze předepsat v dotaze

Koncové body (endpointy)

- Zvláštní typ webových API
- Identifikované pomocí URI
- Používají SPARQL jako komunikační protokol
- Implementovány zpravidla pomocí
  - Nativního úložiště RDF (např. Virtuoso)
  - Relační databáze s nadstavbou pro RDF

Příklady endpointů

- http://dbpedia.org/sparql
  - Pro DBpedii – strukturovanou „verzi“ Wikipedie
- http://dblp.rkbexplorer.com/sparql/
  - Databáze publikací z oblasti informatiky (DBLP)
- Přehled včetně info o dostupnosti (SPARQLES)
  - http://sparqles.ai.wu.ac.at/

SPARQL 1.0

Architektura dotazování

- Dotazový grafový vzor obsahující různé dílčí grafové vzory (zahrnující i filtrační klauzule) se porovnává s obsahem datasetů
- Vždy, kdy lze ve vzoru provést takové dotazení, že se získá určitý podgraf datasetu, vznikne jedno řešení dotazu
- Získaná množina řešení tvoří sekvenci, kterou lze dodatečně upravit pomocí modifikátorů

Grafové vzory (graph patterns)

- Základní GV (basic GP)
- Skupinový GV (group GP)
  - Umístěn v {}
- Nepovinný GV (optional BP)
  - Uveden OPTIONAL
  - Musí být splněný celý; pokud nějaká část splněna není (např. filtr), do výsledků nejde ani splněná část
- Alternativní GV
  - Dva GV spojené UNION
  - Pokud splněno pro oba GV, vygenerují se dvě řešení naráz
- GV nad pojmenovanými grafy
  - GRAPH <graph-IRI> {}
- Dotazový GV – do něj jsou zabaleny všechny GV z dotazu

Datasety

- Dotaz se pokládá proti datasetu
  - Případně i více datasetům: klauzule SERVICE
- Dataset se skládá z
  - právě 1 výchozího grafu (default graph, DG) zkonstruovaného klauzulemi FROM
  - 0 nebo více pojmenovaných grafů (named graphs, NG), každý zkonstruován jednou klauzulí FROM NAMED
- Lze se dotazovat do kteréhokoliv z nich, i v kombinaci
- Časté strategie kombinování grafů:
  - Stejný obsah - plná nebo částečná replikace
  - DG obsahuje metadata ke všem NG, lze je tím vyhledat

Práce s množinou řešení

- Přeuspořádání: ORDER BY
  - Možnost upřesnění vzestupně/sestupně – DESC()
- Projekce
  - Výběr proměnných z grafového vzoru, nebo vše (*)
- Odstranění duplicitních řešení
  - DISTINCT
- Vrácení jen části řešení (stránkování)
  - OFFSET, LIMIT

Typy dotazů vs. množina řešení

- SELECT
  - Vrací přímo jednotlivá řešení v podobě dosazení za proměnné (řádky tabulky, v různých syntaxích – HTML, XML, CSV aj.)
- CONSTRUCT
  - Do výstupního grafového vzoru se postupně dosazují všechna řešení (pro žádné řešení nesmějí ve výstupním grafu zůstat volné proměnné!)
  - Tím vznikne množina grafů, která je nakonec sloučena do jednoho (s odstraněním duplicitních trojic)
- ASK
  - Vrátí hodnotu „pravda“, jakmile se nalezne první řešení
- DESCRIBE
  - Na výstupu graf jako u CONSTRUCT, ale jeho tvar není specifikován v dotazu, jde o „grafové okolí“ množiny prvků ze všech řešení určené algoritmem dané implementace

Filtrování instancí GV

- Klauzule FILTER
  - Je součástí GV, zpravidla až za trojicemi (lze i mezi)
  - Obsahuje výraz nabývající hodnoty pravda/nepravda
- Z množiny řešení jsou vybrána jen ta, pro které je výraz vyhodnocen jako pravdivý
- Výraz je zkonstruován z funkcí a operátorů
  - Některé převzaté z XPath, jiné specifické pro RDF

Klíčové funkce a operátory z FILTER

- *regex*(<vstup>,<regulární výraz>)
  - Třetí argument může být „i“ pro „case-insensitive“
  - Výraz může obsahovat např. „^“ a „ $ “ pro začátek a konec řetězce
- Numerické porovnávání
- Ověřování typu
  - *bound*(), *isIRI*(), *isBlank*(), *numeric*(), *integer*(), …
- Booleovské operace
  - Konjunkce „&&“, disjunkce „||“, negace „!“

SPARQL 1.1

Nové prvky

- Viz příklady ze cvičení, podklad goo.gl/bXX5JY

- Agregační operátory: COUNT, MAX, …
- Poddotazy
  - Dotaz se vyhodnocuje postupně zvnitřku
  - Proměnné z vnitřního dotazu jsou použity ve vnějším
- Dvě varianty negace (doplňku grafového vzoru)
  - FILTER NOT EXISTS, MINUS
- Seskupování: GROUP BY

Nové prvky

- Výpočty výrazů a dosazování do proměnných
  - Při projekci: SELECT <výraz> AS <proměnná>
  - U grafového vzoru: BIND <výraz> AS <proměnná>
    resp. vícenásobně: VALUES <kolekce proměnných>
    {<posloupnost kolekcí n-tic hodnot>}
  - Při seskupování: GROUP BY <proměnná> HAVING <výraz>
- Cesty nad vlastnostmi (property paths)
- Dodatečné funkce a operátory ve FILTER
  - Např. *if*(<výraz>,<když ano>,<když ne>)