Szemantikus adatok lekérdezése federált és osztott rendszereken

Doktori értekezés tézisei

Gombos Gergő

Témavezető: Dr. Kiss Attila

Eötvös Loránd Tudományegyetem
Informatikai Kar
Információs Rendszerek Tanszék

Informatikai Doktori Iskola
Iskolavezető: Prof. Csuhaj-Varjú Erzsébet
Doktori Program: Információs Rendszerek
Programvezető: Prof. Benczúr András

Budapest, 2018

Bevezetés

A Tim-Berners Lee által megalkotott szemantikus web [16] lényege, hogy az interneten olyan formában tároljunk információkat, hogy azok összekapcsolhatóak legyenek egymással, valamint a számítógépek által is értelmezhetőek legyenek. A mai információ keresés főként a Google által kifejlesztett indexelés segítségével és szó alapon történik, amelynek fő hátránya, hogy nem nézi a szavak jelentését csak egyezésüket vizsgálja. Azóta a Google is kifejlesztette a saját tudásbázisát, amely segít az azonos alakú szavak között különbséget tenni. Ennek a tudásbázisnak is az alapja a szemantikus web által megfogalmazott elvárások.

A szemantikus web tárolási struktúráját a W3C az RDF keretrendszerben (Resource Description Framework) [18] fogalmazza meg. Minden fogalomnak és objektumnak rendelkeznie kell egy egyedi azonosítóval, egy úgynevezett IRI-vel (Internationalized Resource Identifier). Ezek az azonosítók a gyakorlatban URL-ek, amelyeknek a kezelése az interneten könnyen megoldott. Ha két adathalmazban szerepel ugyanaz a fogalom, akkor ugyanazt az azonosítót kell használni. Ha megvannak ezek azonosítók, akkor az információkat egyszerű állításokkal írjuk le. Egy állítás alany-állítmány-tárgy felépítésű, és ezeket nevezzük RDF hármasoknak. Egy adott alanyra akár több állítás is megfogalmazható. Az állításokban az állítmány és a tárgy is lehet egy azonosító, emiatt azokra újabb állítások fogalmazhatóak meg. Ezzel a megközelítéssel láthatjuk azt, hogy ezek az állítások úgy is felfoghatóak, mint egy gráf. Az alany és tárgy objektumok a csúcsok, az állítmányok pedig egy címkézett irányított él. Ezeket a gráfokat nevezzük szemantikus gráfnak, és az így tárolt állítások halmazát tudásbázisnak.

A LOD (Linked Open Data) [22] konzorciumnak a célja, hogy az interneten megtalálható információk egy nagy tudásbázisként használhatóak legyenek. Ennek a feltétele, hogy az adathalmazokat úgy kell megadni, hogy az más számára elérhető legyen, valamint kötelező benne olyan azonosítót használni, amely össze tudja kapcsolni az állításokat más adathalmazokkal. Azok az adathalmazok amelyek teljesítik ezeket a követelményeket, kerülnek bele az úgynevezett LOD felhőbe. A LOD felhő azt ábrázolja, hogy egy adathalmaz mekkora, és hogy mennyire kapcsolódik más adathalmazokhoz. A szemantikus adatokat úgynevezett SPARQL végpontokon keresztül lehet elérni. A végpontok egy olyan API-t biztosítanak, amelyek programkódból vagy böngészőn keresztül nyújtanak hozzáférést az adatokhoz.

A szemantikus adatokat a SPARQL lekérdezőnyelv segítségével tudjuk lekérdezni. A SPARQL lekérdezés hasonlít az SQL lekérdezéshez. A *SELECT* után megadjuk mely változók értékeit szeretnénk visszakapni és a *WHERE* részben megfogalmazunk

feltételeket a változókra. A feltételek úgynevezett hármasminták, amelyek olyan hármasok, ahol az egyik adattag egy változó. Az eredmény egy részgráf lesz, amely teljesíti az összes feltételt. Ha több SPARQL végpontról akarunk információt visszanyeri, akkor használhatjuk a SERVICE kulcsszót, melynek megadjuk a végpont elérését és hogy mely hármasmintákat akarjuk ott lefuttatni.
A szemantikus web több ponton is problémába ütközött a megvalósítás terén. A dolgozat ezen problémákkal foglalkozik. Az egyik probléma a hozzáférhetőség. A szemantikus web a felhasználó szempontjából nehezen használható, hisz ismernie kell a szemantikus adathalmazoknak az elérését és az ott található adatokat. Ezekre a problémákra nyújt segítséget a federált rendszerek, amelyek központosítják az adatok elérést. A másik probléma az a szemantikus adatok méretéből, és komplexitásából fakad. A felhasználók nehezen tudják használni a szemantikus adatokat a mobil eszközökön, illetve ha összetettebb lekérdezéseket szeretnének visszakapni. Ezekre a problémákra jelentenek megoldást a kliens-szerver rendszerek és a Big Data eszközök.

Tézisek

1 Szemantikus adatok kliens-szerver architektúrán

A dolgozat 3. fejezete a szemantikus adatok mobilon történő elérésével foglalkozik. A mobilkészülékek gyors fejlődésével és a mobil internet terjedésével lehetőség nyílik a szemantikus adatok használatára ezeken az eszközökön is. Az első mobil készülékek még olyan hardware-rel rendelkeztek, amelyek nem voltak alkalmasak nagy adatok kezelésére. A napjainkba elérhető készülékek már rendelkeznek elég számítási és tárolási kapacitással, de még ez az erőforrás sem lehet elég ha valaki a szemantikus web több millió hármasát szeretné lekérdezni. Másfelől ha ezeken az eszközökön akarjuk a számítást végezni, akkor az jelentősen csökkentené a készülék üzemidejét. Erre jelenthet megoldást a kliens-szerver architektúra, ahol a költséges számítások és az adatok tárolása a szerver eszközökön történik, még a megjelenítés a kliensen. A *'The semantic web: real-world applications from industry'* [20] című könyvben olyan alkalmazási területek találhatóak amelyek az ipar számára is fontosak. Található benne gazdasági, egészségügyi, oktatási vagy akár biztonsági alkalmazások is. Az alkalmazások többsége szerver oldali megvalósítások, amelyeket böngészőn keresztül lehet elérni, de vannak olyan alkalmazások amelyek mobil eszközök számára készültek. Az első és legismertebb ilyen alkalmazás a DBpedia Mobile [21], amely egy hely alapú szemantikus alkalmazás. Ez azt jelenti, hogy felhasználja a

koordinátánkat arra, hogy közelünkben lévő érdekességeket ajánljon nekünk.

Tézis 1 (Egy szemantikus adatokhoz való hozzáférést biztosító szerver alkalmazás lehetőséget ad a vékony klienseknek (mobileszközöknek) a komplex és federált lekérdezések futtatására. [1, 2, 5]). *A mobilkészülékek korlátaik (memória, CPU, adatforgalom, akkumulátor) miatt nem képesek a szemantikus web használatára. Erre lehet megoldás egy olyan kliens-szerver architektúrájú rendszer, ahol a szerver feladata a szemantikus adatok kezelése, lekérdezése, összekapcsolása, a kliens feladata pedig a megjelenítés.*

A dolgozat három alkalmazást mutat be a tézis alapján elkészült rendszerrel. Az első alkalmazás (3.3 fejezet) egy vállalat adatait jeleníti meg mobilkészüléken keresztül szemantikus információk segítségével [1]. A második alkalmazás egy beltéri navigációt [2] (3.4 fejezet) valósít meg. A harmadik alkalmazás pedig egy szemantikus böngésző mobilkészülékek számára [5] (3.5 fejezet).

2 Szemantikus adatok federált környezetben

A 4. fejezet a federált rendszerek témakörével foglalkozik. Ahogy említettem a bevezetésben a szemantikus web használatának fő problémája, hogy a felhasználónak ismernie kell az adathalmaz elérésének az URL-jét, illetve hogy milyen adatok találhatóak ott. Az adatok megismerését nehezíti, hogy a gráfos szerkezet miatt bármilyen információt tárolhatunk az állításokban. A federált rendszerek ebben nyújtanak segítséget. Segítségükkel nincs szükség arra, hogy ismerjük a SPARQL végpontok elérését, vagy hogy ismerjük az ott tárolt adatokat. Az olyan lekérdezéseket amelyek több végpontot is használnak a kiértékeléshez federált lekérdezésnek nevezzük. A federált rendszerek működését mutatja be Rakhmawati és társai [30] a cikkükben. Az federált rendszerben olyan SPARQL lekérdezéseket tudunk megadni, amelyek nem tartalmaznak SERVICE kulcsszót. A végpontok kiválasztása a hármasminták alapján történik. Ennek eldöntésére kétféle megoldás létezik. Az egyik, hogy minden végpontot megkérdezünk hogy meg tudná-e válaszolni az adott hármasmintát (ASK technika), még a másik megoldás információkat tárol a végpontokról (dokumentum technika) és ez alapján dönti el, hogy mely végpont alkalmas a válaszadásra. Annak eldöntését, hogy egy végpont meg tudja-e válaszolni az adott hármasmintát ASK lekérdezésekkel valósítják meg. Ezek a lekérdezések igaz értékkel térnek vissza, ha találnak megfelelő részgráfot, és hamissal ha nem. A katalógus technikát alkalmazza és állítmányokat tárol például a DARQ [19] (distributed ARQ), ASK technikát használ például a FedX [25]. A 4.3. fejezetben az

ASM modell [26] segítségével megfogalmazom a federált rendszerek működését, majd finomítom két konkrét esetre, a szemantikus böngészők és az ajánló rendszerek témakörére. A használatot inspirálta a Grid Rendszerek ASM modellje [17], amely hasonlóan elosztott, párhuzamos működésűek, mint a federált rendszerek.

A szemantikus adatok egyik problémája, hogy a SPARQL lekérdezések megírása nehézkes az adathalmazok ismerete nélkül. Hoefler [28], Campinas [31] és Han [33] is arról ír a cikkében, hogy a szemantikus Web használata bonyolult, mert nehézkes egy SPARQL lekérdezés megírása. A 4.5. fejezetben bemutatok egy olyan megoldást, amely segíti a felhasználókat abban, hogy SPARQL lekérdezéseket készítsenek.

Tézis 2 (**A federált rendszerek segítségével ajánlhatunk hármasmintákat, amely segít a felhasználónak SPARQL lekérdezés elkészítésében. Ennek formális leírása ASM modell-lel valósítható meg.** [4]). *Egy federált rendszer által támogatott ajánló rendszer képes hármasmintákat ajánlani a felhasználónak. Az ajánlások alapja a már részben megírt lekérdezésben szereplő változók. A változó típusa alapján tudunk állítmányokat ajánlani, vagy egy adott állítmány alapján tudjuk a tárgy típusát, amihez újabb ajánlásokat tudunk készíteni. Ez a megoldás segíti a felhasználót a lekérdezések elkészítésében.*

Az ASM modell formálisan írja le a rendszer felé elvárt követelményeinket. A federált rendszerek főbb lépési megfogalmazhatóak az ASM segítségével. A federált rendszerek fő részei: a lekérdezés értelmezés, a végpont kiválasztás, a részlekérdezések futtatása valamint az eredmény elkészítése.

A dokumentum technikára jellemzően a szemantikus adathalmazokban szereplő állítmányokat gyűjtik össze és tárolják az egyes végpontokhoz. A 4.4. fejezetben bemutatok egy olyan megoldást, ahol a szemantikus adatok névterét alkalmazom a végpont kiválasztására.

Mindkét végpont kiválasztási technikának van problémája. Az ASK lekérdezések minden egyes hármasmintánál lekérdezik az összes végpontot. Még a dokumentum alapú technikánál karban kell tartani a végpontokról tárolt információkat. Az ajánló rendszerek segítségével viszont információt kapunk minden végpontról és az ott tárolt adatokról. Erről olvashatunk a 4.6. fejezetben.

Tézis 3 (**Az ajánlásokból kinyerhetőek olyan információk, amelyek támogatják a federált rendszerek végpontválasztását.** [7] **Ilyen információ lehet például a végpontokon található névterek.** [3]). *A SPARQL ajánló rendszer által gyűjtött hármasminták, olyan névtér és állítmány információkat tartalmaznak, amelyek segítik*

a federált rendszer végpont kiválasztási mechanizmusát. A legtöbb ilyen rendszer az adathalmazokban található állítmányokat alkalmazza, de egy adathalmaz jellemzően rendelkezik egy egyedi előtaggal (prefix), amely minden IRI elején megtalálható. Ezek ugyanúgy használhatóak a megfelelő végpont kiválasztására. Az ajánló rendszerből kapott információk alapján, nem szükséges az összes végpontot ellenőrizni a hármasmintáknál. Elég csak azokat vizsgálnunk, amelyről az ajánlatok érkeztek.

3 Szemantikus adatok és az osztott rendszerek kapcsolata

Az 5. fejezet a szemantikus adatok méretéből adódó problémákkal foglalkozik. Az első eredmény amelyről a 5.3. fejezetben írok, kapcsolódik ahhoz, hogy sokszor nem ismerjük a szemantikus adatok szerkezetét, és az abban tárolt adatokat. A vizualizációs eszközök segítik az adathalmaz megértését. Több ilyen vizualizációs eszköz is létezik például: LODmilla [32] vagy a VizBoard [27]. Egy megoldást jelenthet, ha egy gráfmegjelenítő alkalmazás segítségével tudjuk ábrázolni és értelmezni a szemantikus gráfot, viszont a legtöbb ilyen szoftver nem képes kezelni ekkora adatot. A dolgozatban bemutatott módszer a biszimuláció [29] segítségével csökkenti a gráf méretét úgy, hogy a hasonló csúcsokat összevonja. A módszerünket a MapReduce algoritmussal valósítottam meg, amely nagy adatok feldolgozására lett kitalálva.

Tézis 4 (Biszimulációval le lehet csökkenteni a szemantikus gráfokat olyan méretűre hogy azok megjeleníthetőek legyenek gráfmegjelenítő alkalmazásokkal. A biszimulációs algoritmusok megvalósíthatóak MapReduce segítségével, ami a párhuzamos feldolgozást teszi lehetővé. [6]). *Az adatok értelmezését könnyíti, ha az adatokat meg tudjuk jeleníteni valamilyen vizuális formában. A szemantikus adatokat, a méretéből adódóan, a legtöbb gráfmegjelenítő nem tudja kezelni. Erre megoldás a biszimuláció, ami csúcs összevonásokkal képes csökkenteni a gráf méretét. Ahhoz, hogy a gráf méretét csökkenteni tudjuk a Big Data eszközök és a MapReduce programozási modell alkalmazható.*

A szemantikus adatok kapcsolata a Big Data eszközökkel egy másik kutatási irány. Ennek célja, hogy a nagy tudásbázisunkat ezekben az eszközökben tároljuk és valósítsuk meg a lekérdezéseket az adott rendszerhez. Ebben a témában több eredmény született különböző Big Data eszközökkel (Hadoop MapReduce [23], Pig [24]). Ezek az eszközök jellemzően klaszterben képesek hatékonyan működni, az oszd-meg-és-uralkodj elvhez ha-

sonlóan. A probléma ezekkel, hogy sok lemezműveletet használnak, amelyek lassítják a feldolgozást. A Spark viszont egy új adattárolást valósított meg, amely a klaszterben lévő gépek memóriájában tárolja az adatokat, ezzel hatékonyabb tud lenni. A másik előnye a Sparknak, hogy rendelkezik olyan könyvtárral, ami gráf elemzésekre szolgál. Mivel a szemantikus adatok gráfként is felfoghatóak, így az osztott gráfelemzés módszerével valósítottam meg a SPARQL kiértékeléseket. A 5.4. fejezetben bemutatok egy módszert arra, hogyan lehet a szemantikus adatokat tárolni egy elosztott gráf rendszerben, illetve bemutatok egy lekérdezési tervet, amelyet a Spark GraphX fog alkalmazni. Az osztott gráf elemzés alapja, hogy a csúcsok üzenetet küldenek a szomszédaiknak az élek mentén. A lekérdezési terv azt adja meg, hogy mely élek és milyen irányban küldhetnek egy adott iterációban üzenetet. Eredményeimet összevetettem a Spark GraphX-en futó S2X [34] rendszerrel, amelyet Schätzle és társai készítettek.

Tézis 5 (**SPARQL lekérdezési modell Spark GraphX rendszerhez lineáris lekérdezési tervvel (Spark(q)l) [8]).** *A Spark GraphX elosztott gráfelemző rendszer használható a szemantikus adatok gráfként történő tárolására és lekérdezésére. A lineáris lekérdezési terv mindig egy adott hármasmintát értékel ki egy adott iterációban. Ahhoz, hogy ez működni tudjon, meg kell határoznunk egy sorrendet az élek között, és csak azokat a részeredményeket fogadjuk el, amelyeket az előző iterációban számoltuk ki. A lekérdezés kiértékelésekor nem tudjuk, hogy egy adott állítmány Data property-e, emiatt hurok élekkel tudjuk ellenőrizni, hogy az adott tulajdonság szerepel-e a csúcs adatai között. A GraphX működéséből adódóan lehetnek olyan élek, amelyek nem érünk el a kiértékelés során, hisz az előző iterációkban nem használtuk őket. Emiatt 'alive' üzenetekre van szükség, mely életben tartja az élt addig, amíg fel nem dolgozzuk.*

A 5.5. fejezetben bemutatom az algoritmus javított verzióját, amely párhuzamos lekérdezési tervvel képes a kiértékelést elvégezni.

Tézis 6 (**SPARQL lekérdezési modell Spark GraphX rendszerhez párhuzamos lekérdezési tervvel (P-Spark(q)l) [9]).** *A Spark GraphX-en történő SPARQL kiértékelést úgy is lehet végezni, hogy egyszerre több hármasmintát értékelünk egy adott iterációban. Ekkor nincs szükségünk olyan üzenetekre, amelyek életben tartják az éleket a kiértékelésig, hisz a kiértékelési fa egy adott szintjén az élek aktívak lesznek. Cserébe viszont az üzenetek összekapcsolása bonyolultabb, mivel a sémájuk nem egységes. Egyes üzeneteket uniózni és még másokat összekapcsolni kell egy adott iterációban. A hurok élek problémájára a gráf beolvasásakor gyűjthetünk információt. Meg tudjuk állapítani mely állítmányok Data propertyk, illetve összegyűjthetjük az Objekt propertyk számosságát, amivel az iterációk sorrendjét változtathatjuk a hatékonyság növelése érdekében.*

Szerző publikációi

[1] **Gergő Gombos** et al. “A Mobile browser prototype for semantic infromation systems”. In: *Acta Electrotechnica et Informatica* 13.4 (2013), pp. 20–25.

[2] Tamás Matuszka, **Gergő Gombos**, and Attila Kiss. “A New Approach for Indoor Navigation Using Semantic Webtechnologies and Augmented Reality”. In: *Virtual Augmented and Mixed Reality. Designing and Developing Augmented and Virtual Environments*. Springer, 2013, pp. 202–210.

[3] **Gergő Gombos** and Attila Kiss. “SPARQL Processing over the Linked Open Data with Automatic Endpoint Detection”. In: *Advanced Approaches to Intelligent Information and Database Systems*. Springer, 2014, pp. 183–192.

[4] **Gergő Gombos** and Attila Kiss. “SPARQL Query Writing with Recommendations Based on Datasets”. In: *Human Interface and the Management of Information. Information and Knowledge Design and Evaluation*. Springer International Publishing, 2014, pp. 310–319.

[5] Tamás Matuszka, **Gergő Gombos**, and Attila Kiss. “mswb: Towards a mobile semantic web browser”. In: *Mobile Web Information Systems*. Springer, 2014, pp. 165–175.

[6] Gábor Rácz, **Gergő Gombos**, and Attila Kiss. “Visualization of Semantic Data Based on Selected Predicates”. In: *Transactions on Computational Collective Intelligence XIV*. Springer, 2014, pp. 180–195.

[7] **Gergő Gombos** and Attila Kiss. “Federated Query Evaluation Supported by SPARQL Recommendation”. In: *International Conference on Human Interface and the Management of Information*. Springer. 2016, pp. 263–274.

[8] **Gombos, Gergő**, Gábor Rácz, and Attila Kiss. “Spar (k) ql: SPARQL evaluation method on Spark GraphX”. In: *Future Internet of Things and Cloud Workshops (FiCloudW), IEEE International Conference on*. IEEE. 2016, pp. 188–193.

[9] **Gombos, Gergő** and Attila Kiss. “P-Spar(k)ql: SPARQL Evaluation Method on Spark GraphX with Parallel Query Plan”. In: *2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*. IEEE. 2017, pp. 212–219.

Szerző további publikációi

[10] Antal Iványi et al. “Parallel enumeration of degree sequences of simple graphs II”. In: *Acta Universitatis Sapientiae, Informatica* 5.2 (2013), pp. 245–270.

[11] Antal Iványi et al. “Score sets in multitournaments I. Mathematical results”. In: *ANNALES UNIVERSITATIS SCIENTIARUM BUDAPESTINENSIS DE ROLANDO EOTVOS NOMINATAE SECTIO COMPUTATORICA*. Vol. 40. ELTE. 2013, pp. 307–320.

[12] Imre Szucs, **Gergő Gombos**, and Attila Kiss. “Five Ws, one H and many tweets”. In: *Cognitive Infocommunications (CogInfoCom), 2013 IEEE 4th International Conference on*. IEEE. 2013, pp. 441–446.

[13] **Gergő Gombos** et al. “VOSD: A General-Purpose Virtual Observatory over Semantic Databases.” In: *Acta Cybern.* 21.3 (2014), pp. 353–366.

[14] **Gergő Gombos**, Attila Kiss, and Zoltán Zvara. “Performance Analysis of a Cluster Management System with Stress Cases”. In: *Acta Polytechnica Hungarica* 13.2 (2016).

[15] Sándor Laki et al. “Take your own share of the PIE”. In: *Applied Networking Research Workshop (ANRW)*. 2017.

További publikációk

[16] Tim Berners-Lee, James Hendler, Ora Lassila, et al. “The semantic web”. In: *Scientific american* 284.5 (2001), pp. 28–37.

[17] Zsolt N Németh and Vaidy Sunderam. “A formal framework for defining grid systems”. In: *Cluster Computing and the Grid, 2002. 2nd IEEE/ACM International Symposium on*. IEEE. 2002, pp. 202–202.

[18] Graham Klyne and Jeremy J Carroll. “Resource description framework (RDF): Concepts and abstract syntax”. In: (2006).

[19] Bastian Quilitz. *DARQ–Federated Queries with SPARQL*. 2006.

[20] Jorge Cardoso, Martin Hepp, and Miltiadis D Lytras. *The semantic web: real-world applications from industry*. Vol. 6. Springer Science & Business Media, 2007.

[21] Christian Becker and Christian Bizer. “DBpedia Mobile: A Location-Enabled Linked Data Browser.” In: *Ldow* 369 (2008), p. 2008.

[22] Christian Bizer, Tom Heath, and Tim Berners-Lee. "Linked data-the story so far". In: *Semantic services, interoperability and web applications: emerging concepts* (2009), pp. 205–227.

[23] Mohammad Farhan Husain et al. "Storage and retrieval of large rdf graph using hadoop and mapreduce". In: *Cloud computing*. Springer, 2009, pp. 680–686.

[24] Alexander Schätzle, Martin Przyjaciel-Zablocki, and Georg Lausen. "PigSPARQL: Mapping SPARQL to Pig Latin". In: *Proceedings of the International Workshop on Semantic Web Information Management*. ACM. 2011, p. 4.

[25] Andreas Schwarte et al. "FedX: Optimization techniques for federated query processing on linked data". In: *The Semantic Web–ISWC 2011*. Springer, 2011, pp. 601–616.

[26] Egon Börger and Robert Stärk. *Abstract state machines: a method for high-level system design and analysis*. Springer Science & Business Media, 2012.

[27] Martin Voigt, Stefan Pietschmann, and Klaus Meißner. "Towards a semantics-based, end-user-centered information visualization process". In: *Proc. of the 3rd international workshop on semantic models for adaptive interactive systems (SEMAIS 2012)*. 2012.

[28] Patrick Hoefler. "Linked Data Interfaces for Non-expert Users". In: *The Semantic Web: Semantics and Big Data*. Springer, 2013, pp. 702–706.

[29] Yongming Luo et al. "Bisimulation reduction of big graphs on mapreduce". In: *British National Conference on Databases*. Springer. 2013, pp. 189–203.

[30] Nur Aini Rakhmawati et al. "Querying over Federated SPARQL Endpoints—A State of the Art Survey". In: *arXiv preprint arXiv:1306.1723* (2013).

[31] Stéphane Campinas. "Live SPARQL Auto-Completion". In: *ISWC 2014 Posters & Demonstrations Track*. CEUR-WS.org, 2014, pp. 477–480.

[32] András Micsik, Sándor Turbucz, and Zoltán Tóth. "Browsing and Traversing Linked Data with LODmilla." In: *ERCIM News* 2014.96 (2014).

[33] Lushan Han et al. "Querying RDF data with text annotated graphs". In: *Proceedings of the 27th International Conference on Scientific and Statistical Database Management*. ACM. 2015, p. 27.

[34] Alexander Schätzle et al. "S2X: Graph-Parallel Querying of RDF with GraphX". In: *Proc. of 1st International Workshop on Big-Graphs Online Querying*. 2015.