Semantic Web in the Fog of Browsers

Pascal Molli and Hala Skaf-Molli

University of Nantes, LS2N, France
{pascal.molli,hala.skaf}@univ-nantes.fr

Abstract. Imagine connecting thousands of web browsers with browser-to-browser connections, sharing storage, bandwidth, and CPU. This builds a fog of browsers where end-user devices are ready to collaborate. Imagine semantic fog applications running in fogs of browsers, querying the linked data servers hosted in the cloud and data hosted in the fog. Fogs of browsers running semantic fog applications create a new massively decentralized infrastructure where RDF data and SPARQL query processing are available both on web servers and on browsers. In this paper, we explore new opportunities and research challenges opened by a fog of browsers for the semantic web.

1 Introduction

Fog computing relies on the collaboration of a multitude of devices located near end-users to provide new services or improve cloud services [14]. There are interdependencies between fog computing and with cloud computing. Fog can act as a proxy to improve the quality of services of cloud services. Moreover, fog can be the beachhead of the cloud to collect and aggregate data [5]. Cloudlets [12], Cisco IOx, or Paradrop are good examples of how the fog nodes can be implemented and deployed near end-user devices [15].

In the context of the semantic web, fog computing is able to improve the availability of semantic data without increasing the cost of hosting data for data providers. It could also greatly help the semantic web to aggregate new data collected from end-users or from the web of things.

Traditionally, fog computing uses network gateways to run fog nodes. In the context of the semantic web, we believe that web browsers also meet naturally most of the criteria for fog computing. Web Browsers are located near end-users, they have storage, CPU, communication, and most of all, they are de facto the most widely deployed execution environments in the world. The recent introduction of WebRTC [1] has further extended the capabilities of browsers by introducing support for browser-to-browser communications. This turns browsers into a decentralized execution environment for running semantic web applications. As browsers have already the ability to locally store RDF data and in some context to run SPARQL queries, semantic fog applications running in the fogs of browsers create a new massively distributed infrastructure where RDF data and SPARQL query processing are available both on the cloud and the fog.

Therefore, the main challenge is to decide how to locate RDF data and process SPARQL queries over this massively distributed infrastructure to deliver services and quality of services required for a given semantic web application.

In this paper, we explore new opportunities and research challenges opened by a fog of browsers for the semantic web.

[1] https://webrtc.org/

The paper is organized as follows. Section 2 defines semantic fog applications in the fog of browsers. Section 3 presents new opportunities opened by semantic fog applications. Section 4 highlights new research challenges for semantic fog applications. Finally, conclusions are outlined in section 5.

2 Semantic Fog Applications in the Fog of Browsers

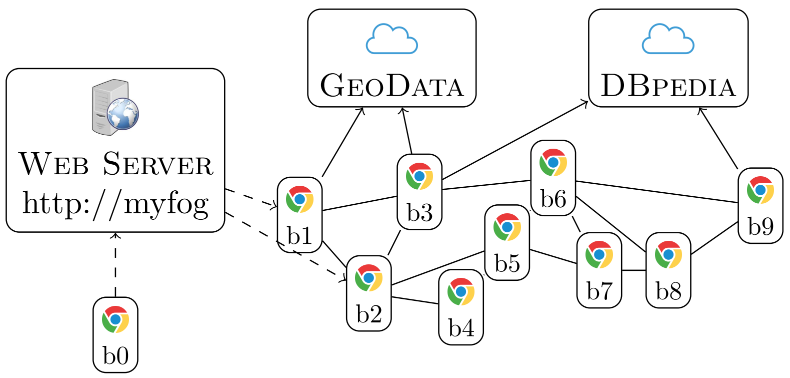

Fig. 1. A semantic fog application running in a fog of browsers

A fog of browsers is a set of interconnected browsers with browser-to-browser connections. Such connections are now supported thanks to the WebRTC standard in Firefox, Chrome, Microsoft Edge and IOS. A browser can participate to one or several fogs.

A fog of browsers is accessible through one or several URIs hosted on regular web server. The web server dereferences this address to a JavaScript application bootstrapped with a sample of already connected browsers[2]. This JavaScript application represents a semantic fog application with its own logic. The application is able to manage RDF data and runs SPARQL queries over linked data and/or over data hosted in the fog. We assume that all RDF data are managed following the linked data principles [3].

Once downloaded in the browser, the semantic fog application joins the network of browsers by connecting the browser to at least one of the already connected browsers. Following this approach, at a given time, there is potentially a high number of browsers, running the same application, and all these browsers are connected together. We do not make any assumption about the topology of the network, i.e. hierarchical, structured, unstructured, hybrid or multi-layer. Topology depends on the objective of the semantic fog application. In figure 1, the browsers are connected in an unstructured network, they execute SPARQL queries over data in the fog and 2 datasources hosted in the cloud. The browser $b0$ contacts the web server hosting the semantic fog application in order to join the network. The web server is returning the semantic fog application and references to two browsers: $b1$ and $b2$. $b0$ contacts one of them to join this fog.

To be usable, a semantic fog application must meet the following requirements inspired from P2P data management [11]:

[2] As has been already done in [9] and [7].

autonomy Each browser participating in a fog of browsers is free to join and leave at any time. It owns its data and have a full control on it.

query expressiveness A semantic fog application runs SPARQL queries or a subset of SPARQL. The scope of the query can refer traditional linked data providers and/or fog participants.

efficiency A fog of browsers is composed by the resources of fog participants and the resources of cloud providers involved in the semantic fog application. The efficient uses of all resources should result in higher throughput of queries.

quality of service The fog has to improve the user-perceived efficiency of the system.

fault tolerance Quality of service can be maintained for a period of time even in presence of failures of browsers or failures of linked data providers.

security As an open system, a fog of browsers can be used to steal personal data, attack other browsers in the fog or attack servers. Access control and resistance to malicious are crucial for semantic fog applications.

3 Semantic Fog Applications

Deploying semantic fog applications over a fog of browsers raises several opportunities. In this section, we present several semantic fog applications illustrating different usages.

3.1 Queries in the fog

```
void main() {

/* Connect browsers geolocalized in Nantes */
 Overlay.configure(Geolocation='Nantes');

/* Q1: Get nearby point of interest */
 String query= 'select ?place where ?place nearby ('+myposition+' 100m)';
 ResultSet  loc = queryExecution.exeSelect(query,model)

/* Q2: Collect user feedback */
 answer=askUser('Do you like your location?:'+loc)
 UpdateAction.execute('INSERT DATA'+'me'+op:likes+ loc  );

 /* Q3: Display most liked places */
 queryExecution.exeSelect(
    'SELECT ?place COUNT(?likes) { ?place liked ?o} groupby ?place'))
}
```

Fig. 2. The "tourism in Nantes" semantic fog application

Consider a simple semantic web application where people visiting a city have access to point of interests around them, can rate these points, and can list top-ranked point of interests. This application can be written with queries like $Q1$, $Q2$, and $Q3$ presented in figure 2 and can be deployed in the cloud. Consequently, queries are executed in the cloud with data stored in the cloud datastore. In this case, the cost of running this application relies entirely on the application provider, the availability and the performances of the application relies on the cloud provider.

Now consider that the code of the Figure 2 is a semantic fog application that is loaded and run in each browser visiting the web page of this application. The line 4, the semantic fog application connects the browser of the visitor to a fog of browsers where browsers are now located in the city of Nantes.

Concerning query $Q1$, data are still located in the cloud, but now the fog could provide data caching and consequently could improve data availability and reduce the cost of data providers. Under certain conditions, the application could continue to run even if cloud services are unavailable.

Concerning query $Q2$, the semantic fog application can be configured to store data locally in the browser, in the fog, or in the cloud. Suppose user feedback is stored locally in the browser. Then the cost of executing $Q2$ is no more on the charge of the application provider and furthermore, user ratings do not leave browsers.

Query $Q3$ is dependent on data location. Different situations are possibles: if users ratings are stored in the cloud, then executing $Q3$ in the fog is similar to $Q1$. If users ratings are stored in their own browsers, then $Q3$ execution requires to contact every browsers. If users ratings are stored somewhere in the fog, then data can be smartly located and aggregated to answer $Q3$ efficiently without contacting every browsers.

As we can see, running semantic fog application opens different trade-off concerning what can be done in the fog and what can be done in the cloud. This trade-off impacts the cost of running an application, the performance of the application, and the availability of the application. It can also impact the privacy of personal data and the quality of collected data.

3.2 Semantic Collaborative caching

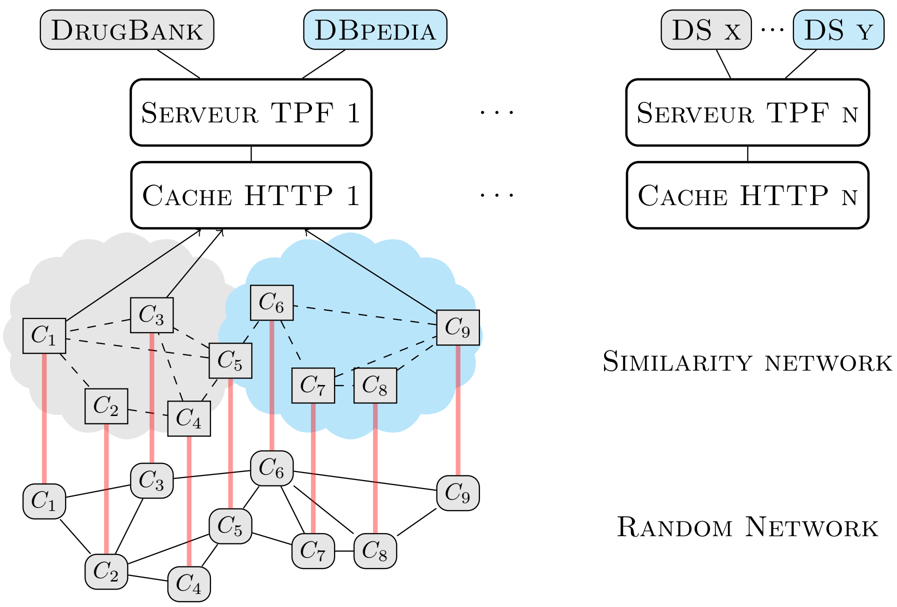

Fig. 3. Semantic Collaborative caching

Cyclades [6] is a collaborative caching system that can be used by a semantic fog application as the one presented in figure 2. Cyclades connect similar browsers by assuming that users with similar queries in the past will certainly perform similar queries in the future. Therefore, data cached at similar nodes could be used to answer queries without using resources of linked data servers.

Cyclades is based on a double overlay networks; the first one builds a random network providing connectivity while the second one incorporates a similarity metric. The similarity metric is able to detect users performing similar queries based on the analysis on their local caches. The two-level network topology of Cyclades is described in figure 3.

In this scenario, the fog is able to reduce the number of calls to data providers. Consequently, this improves data availability and reduces the cost of providing data.

3.3 Queries with the fog

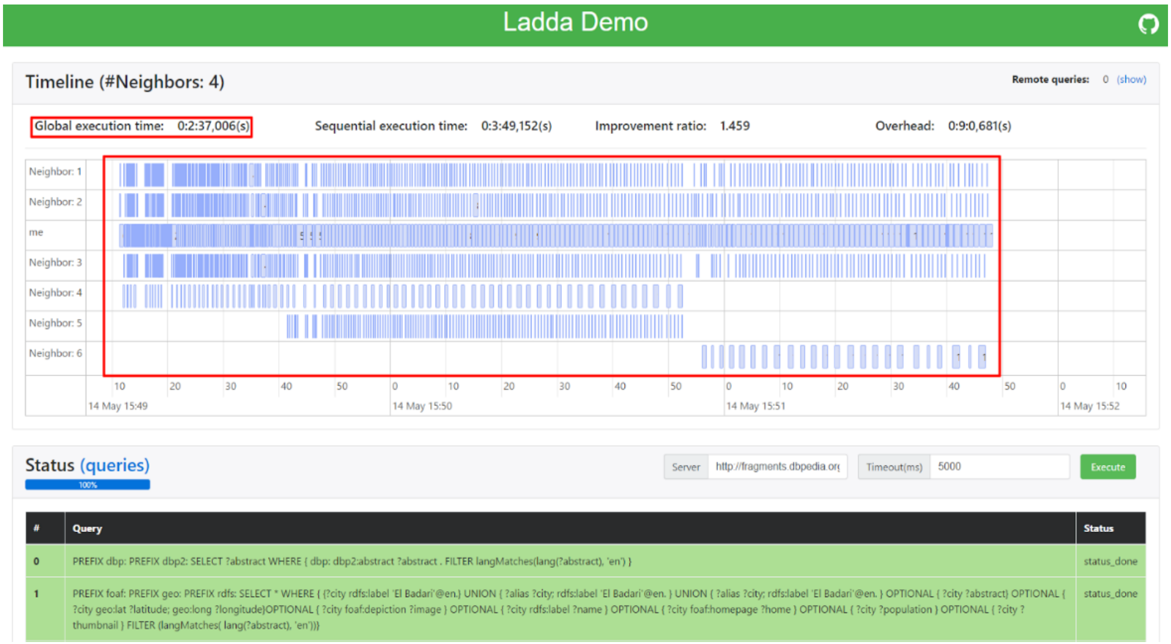

Fig. 4. Queries with the fog

Ladda [7] is a semantic fog application that allows participants to delegate their SPARQL queries to their neighbors in the fog. For example, one can want to execute:

```
for each $country in countries
query.execute("SELECT ?software ?company WHERE {
  ?software dbpedia-owl:developer ?company.
  ?company dbpedia-owl:locationCountry
      [ rdfs:label "$country"@en ].
}'
```

By parallelizing the execution of queries over different browsers, the execution time of this workloads can be significantly reduced. Figure 4 illustrates a Ladda's query execution.

In this execution, a browser executes 1509 queries with the help of 6 neighbors in a network composed of 50 participants. Each square represents the execution time of a query on the swim lane of a browser. On this run, the execution time of the workload is 2m37s instead of 3m32s if the workload was executed by one browser.

In this scenario, the semantic fog application allows to share the CPU and bandwidth of browsers for SPARQL query processing.

4 Research challenges

Deploying semantic fog applications on a fog of browsers opens many opportunities for semantic web application developers. They can optimize financial costs, availability, performances, privacy . . . However, the programming model has to remain simple as the one depicted in Figure 2. The configuration of the semantic fog application has to determine how queries and data are deployed in the cloud and in the fog to reach developper expectations.

The fog of browsers can reuse some scientific results from P2P data management systems [11, chapter 16]. Many works demonstrated how data can be efficiently stored and accessed on structured, unstructured, and hybrid P2P networks such as Edutella [10], RDF-Peers [4], PierDB [8], GridVine [1] etc. However, the context and objectives of fog of browsers are slightly different:

- Fog and cloud are interdependent. Cloud services can be used to manage the fog. The fog can just improve the efficiency and the quality of services of data providers without managing data as demonstrated in [6] and [7].
- Most of work on P2P data management have been done on TCP/IP networks. However, WebRTC networks used by browsers have several major differences with traditional TCP/IP networks:
  1. A WebRTC network is not addressable and basically has no routing. Consequently, contacting a particular browser can be costly.
  2. Establishing a WebRTC connection between 2 browsers requires a third party to exchange tokens. Once tokens exchanged, a complex negotiation protocol starts to allow NAT traversal. So, establishing a WebRTC connection can be more costly than a TCP/IP connection.

  The constraints of WebRTC change the cost of communications and potentially impact all existing algorithms.

Customized overlay networks for a fog of browsers. A fog of browsers connects thousands of browsers over WebRTC. The nature of WebRTC networks and the objective of the semantic fog application can lead to different design choices. As routing is costly in WebRTC, keeping useful neighbors around us in one hop, can be a good strategy for efficiency and quality of service. Indeed, direct neighbors can be contacted at low cost. 'useful neighbors' can have different meanings according the application. Many similarity metrics can be defined and many overlay can be combined in the same fog as proposed in [6]. Finding the best similarity metrics, topologies and combinations of topologies for query efficiency and quality of services is clearly an important research direction.

Dynamic replication and consistency in a fog of browsers. Data replication is a fundamental concept for improving data availability and performances of query processing. In the context of a fog of browsers, replication contributes to query efficiency, quality of service and fault-tolerance requirements. A replication strategy has to decide what data to replicate, where to replicate and when to replicate. Such decisions are complex in a fog of browsers: the participants are autonomous, the data storage is limited, the communication costs constrained by network topology. Adaptivity of replication to queries seems a good strategy. Materializing data fragments that are frequently retrieved from data providers and spreading them within the fog can have a significant impact on performances. Defining these fragments, deciding when to replicate them and where to locate them is clearly challenging. Another challenge strongly related to data replication is consistency management. Data needs to be up-to-date. Maintaining consistent data fragments at low-cost in a fog of browsers is clearly challenging.

Crowdsourcing with a fog of browsers A browser is not just an execution environment for JavaScript programs. It could also involve humans with their Web of Things devices. Fog computing allows a collaboration between man and machines to collect, curate and aggregate data. Consequently, a fog of browsers can be seen as a distributed crowdsourcing platform where data are collected, semantified and verified within the fog, before saved to the cloud. How the functionalities of a crowdsourcing platform can be distributed among the fog and cloud providers is an interesting challenge.

Federated query engines for a fog of browsers. Federated SPARQL query engines [13, 2] allow to query several data sources in a transparent way. In the context of a fog a browsers, the fog itself could be considered as a new data source that cloud be combined with traditional data providers. However, each fog participant has a fragment of data and has to be contacted to answer queries. Such problems have been partially addressed by P2P data management systems. The challenge is to build a distributed federated query engine running in the fog, able to query data in the cloud and in the fog.

Security for semantic fog application If a fog of browsers opens many opportunities, it also brings new threats: A fog of browsers can be used to perform DDOS attacks, to steal personal information from browsers, and to watch people. A semantic fog application has to protect participants and data providers against malicious users. Semantic fog applications require appropriate security models.

5 Conclusions

In this paper, we presented how semantic fog applications running in the fog of browsers creates a massively decentralized infrastructure that extends the semantic web to the browsers of end-users. By this way, the semantic web can take advantage of resources of browsers, including end-users and IoT devices. Semantic fog applications can improve the efficiency and quality of service of linked data providers. It can also enhance the linked data with data provided by end-users.

If some semantic fog applications are already there, more research efforts are needed to fully exploit all the potential of semantic fog applications: pertinent network topologies, dynamic replication, efficient query processing, data quality and security.

Another interesting research questions have not been discussed in this paper: the dynamicity of the fog of browsers and how fog of browsers can be combined with distributed ledgers for commercial query processing in the fog of browsers.

References

1. Karl Aberer, Philippe Cudré-Mauroux, Manfred Hauswirth, and Tim Van Pelt. Gridvine: Building internet-scale semantic overlay networks. In *International semantic web conference*, volume 3298, pages 107–121. Springer, 2004.
2. Maribel Acosta, Maria-Esther Vidal, Tomas Lampo, Julio Castillo, and Edna Ruckhaus. Anapsid: an adaptive query processing engine for sparql endpoints. *The Semantic Web–ISWC 2011*, pages 18–34, 2011.
3. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *International Journal of Semantic Web and Information Syststems*, 5(3):1–22, 2009.
4. Min Cai and Martin Frank. Rdfpeers: a scalable distributed rdf repository based on a structured peer-to-peer network. In *Proceedings of the 13th international conference on World Wide Web*, pages 650–657. ACM, 2004.
5. Mung Chiang and Tao Zhang. Fog and iot: An overview of research opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.
6. Pauline Folz, Hala Skaf-Molli, and Pascal Molli. CyCLaDEs: a decentralized cache for Linked Data Fragments. In *ESWC: Extended Semantic Web Conference*, 2016.
7. Arnaud Grall, Pauline Folz, Gabriela Montoya, Halla Skaf-Molli, Pascal Molli, Miel Vander Sande, and Ruben Verborgh. Ladda: SPARQL queries in the fog of browsers. In *Proceedings of the 14th ESWC: Posters and Demos*, May 2017.
8. Ryan Huebsch, Joseph M Hellerstein, Nick Lanham, Boon Thau Loo, Scott Shenker, and Ion Stoica. Querying the internet with pier. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 321–332. VLDB Endowment, 2003.
9. Brice Nédelec, Pascal Molli, and Achour Mostefaoui. Crate: Writing stories together with our browsers. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 231–234. International World Wide Web Conferences Steering Committee, 2016.
10. Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjörn Naeve, Mikael Nilsson, Matthias Palmér, and Tore Risch. Edutella: a p2p networking infrastructure based on rdf. In *Proceedings of the 11th international conference on World Wide Web*, pages 604–615. ACM, 2002.
11. M Tamer Özsu and Patrick Valduriez. *Principles of distributed database systems -*. Springer, 2011.
12. M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4):14–23, Oct 2009.
13. Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. Fedx: Optimization techniques for federated query processing on linked data. In *International Semantic Web Conference*, pages 601–616. Springer, 2011.
14. Luis M Vaquero and Luis Rodero-Merino. Finding your way in the fog: Towards a comprehensive definition of fog computing. *ACM SIGCOMM Computer Communication Review*, 44(5):27–32, 2014.
15. Shanhe Yi, Zijiang Hao, Zhengrui Qin, and Qun Li. Fog computing: Platform and applications. In *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on*, pages 73–78. IEEE, 2015.