MoRAI: Geographic and Semantic Overlay Network for Linked Data Access with Intermittent Internet Connectivity

Mahamadou Toure, Kaladzavi Guidedi, Fabien Gandon, Moussa Lo, Christophe Guéret

MoRAI: Geographic and Semantic Overlay Network for Linked Data Access with Intermittent Internet Connectivity

Mahamadou Toure
University Côte d'Azur
and Gaston Berger University
Nice, France and St-Louis, Senegal
mahamadou.toure@inria.fr

Kaladzavi Guidedi
University of Maroua
Maroua, Cameroon
kaladzavi@univ-maroua.cm

Fabien Gandon
INRIA, University Côte d'Azur
Sophia Antipolis, France
fabien.gandon@inria.fr

Moussa Lo
Universite Virtuelle du Senegal
Dakar, Senegal
moussa.lo@uvs.edu.sn

Christophe Gueret
Accenture Labs
Dublin, Ireland
christophe.gueret@accenture.com

***Abstract*—We propose and evaluate MoRAI (Mobile Read Access in Intermittent internet connectivity), a distributed peer-to-peer architecture organized in three levels dedicated to RDF data exchanges by mobile contributors. We present the conceptual and technical aspects of this architecture as well as a theoretical analysis of the different characteristics. We then evaluated it experimentally and results show the relevance of considering geographical positions during data exchanges and of integrating RDF graph replication to ensure data availability in terms of requests completion rate and resistance to crash scenarios.**

***Index Terms*—linked data, peer-to-peer, overlay network, limited access**

I. Introduction

Peer-to-peer (P2P) model has positioned itself as an effective approach to meet the dynamism and scalability requirements of distributed systems. To overcome constraints related to the availability and reliability of sources of data, P2P systems rely on mechanisms to mitigate the impact of disconnection scenarios on data availability, but also to balance the system's processing loads on all peers.

Our work builds on a joint interest in combining semantic Web approaches with P2P Web overlay networks: (1) overlay networks provide more resilient and efficient connectivity for Web agents to exchange their data and, in return, (2) Web formalisms and protocols provide the needed standards to ensure interoperability between heterogeneous peers, other architectures and heterogeneous resources. We propose MoRAI (Mobile Read Access in Intermittent), an architecture dedicated to RDF data exchange through mobile applications. MoRAI allows users to access and share data in environments where hardware resources are limited and Internet access is intermittent. Intermittent Internet access refers here to cases where the users frequently only have a local wifi or data connection with TCP/IP but do not have access to remote Internet resources especially outside the country or continent. In this case users continue to have local access but the failure of a national or international link prevents them from accessing external resources such as DBpedia or Wikidata for example.

In this context, MoRAI allows locally connected systems to continue operating. In this P2P architecture, peers are applications running in a Web browser. To connect to the linked data sharing network the user just has to open the Web application to become a new peer.

This paper details the following contributions:

1) MoRAI, a peer-to-peer logical architecture dedicated to environments with limited Internet access and hardware resources and ensuring participant connectivity and data availability.
2) A geographical overlay network based on peer locations to consider geographically close neighbours and combined with a semantic overlay in order to provide each participant with a relevant and available knowledge base.
3) Experimental results showing the relevance of: a) considering geographical positions during exchanges and b) integrating RDF graph replication. Results show the impact of these two points on the completion rate and the system's resistance to crash scenarios.

This paper is organized as follows: section II introduces our motivating scenario. Section III positions our contribution in the state of the art. Sections IV and V respectively present the model and algorithms details. We present the evaluation settings and results in section VI and conclude in Section VII.

II. Motivating scenario: sharing at a local event

We aim at scenarios where users participate to an event at a given location particularly characterized by intermittent internet access. During this event, participants represented by browsers on their mobiles form a virtual peer-to-peer network.

A real example of such a scenario that motivated our work is the International Jazz Festival of Saint-Louis in Senegal. During this event, the constant need to access and share information related to the artists, the schedule, the locations and the contents is an important point for improving the participants' experience. More precisely, each user must have the ability to exchange data with semantically close neighbours. For example, in the case of the Jazz Festival, some participants will be interested by data related to music such as concerts and others by data related to cultural activities such as art expositions. Participants must also be able to exchange with geographically close neighbors. The idea is that, during data exchange operations, several peers will be geographically near their points of interest. In the case of the Jazz Festival for example, we assume that a number of the peers requesting information about expositions will be near the art museum.

III. Related Work

In CyCLaDEs [3], When a request is processed by a given client, each sub-request pattern triple is searched first on the local cache, then in the cache of its neighbors and finally on the LDF server if necessary. CyCLaDEs relies on a random peer sampling architecture for member composition management and a clustered architecture to handle the k-best neighbors.

FireChat [4] relies upon open-access mesh network and is able to use phone-to-phone bluetooth signals to connect whenever mobile phone coverage is unavailable. However, FireChat does not give users the ability to query their neighbors, or to access data preceding its arrival in the community.

Solid [5] user's data is stored in a Web-accessible personal online datastore (pod). To share information with others (individuals or organisations), user give permission to access the appropriate information in his pod. Querying several users' pods however remains an open question on Solid.

[9] shows that in traditional DTN there is a balance to be found between the message delivery rate and the network load in terms of messages. However, a high message load means extra energy costs for mobile devices.

In Snob [1], direct neighbours in the network are the data sources and results received from neighbours are stored locally as intermediate results. To speed up query execution, browsers processing similar queries are connected through a semantic overlay network.

In this paper we combine and extend on the one hand the Cyclades approach regarding the hierarchy of requests processing : first in the local datastore, then in the neighbors' datastore and finally in the super-peers' datastores and on the other hand, the MoRAI request execution model is close to the Snob model. MoRAI allows users to query several data sources at once. The difference comes from taking into account the location of the neighbours to increase the search perimeter and super-peer integration on which the data graph is replicated. This allows us to improve data availability, thus providing the system with a better completion rate. Our approach may remind of the cellular network with base stations, however this type of network is less efficient than the wifi network used here in terms of data transmission rates and requirements on mobile devices. Instead, MoRAI would be more comparable to the concept of edge computing in the sense that mobile devices transfer part of their request processing task to super-peers during first executions. Nevertheless, the super-peers act here as a stater of the architecture and will be less and less used during execution.

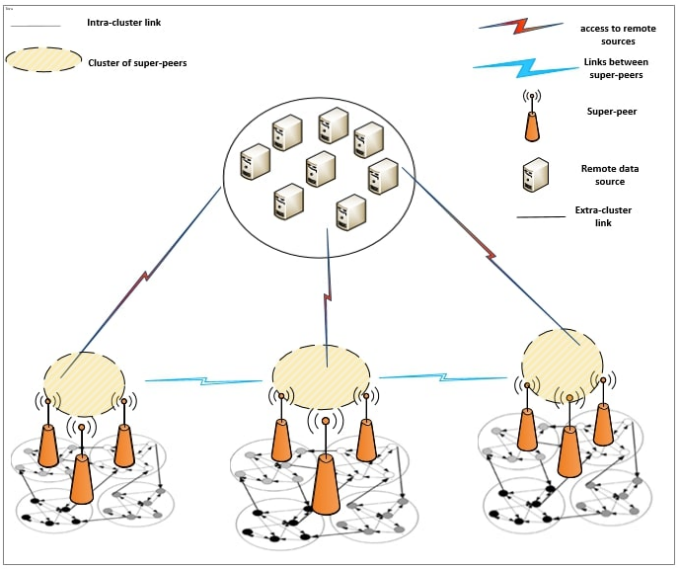

Fig. 1. MoRAI Architecture: mobile peers (black dots), super-peers and remote sources.

IV. MoRAI semantic & geographic overlay model

Our logical architecture is adapted to constraining environments and allows mobile contributors to share locally, via a browser network, an RDF dataset. It consists of 3 levels (Fig. 1): single peers, super peers and remote sources.

1) Peers: we consider each browser as a peer participating in the architecture and they form the level 1. To each peer is attached a profile defined as a set of triple pattern of queries. The profile is built with the ability to store several requests which can represent several interests. Each peer also holds a local triple store containing RDF triples. As these peers represent web browsers, RDFStore-js [6] is used as a local triple store and query engine. Each browser host an RDFStore-js and be able to integrate MoRAI as a NodeJs or JavaScript client. To take mobility in consideration, a pair of geographic coordinates is assigned to all peers and is used for calculating the Euclidean distance between them.

Requests are first processed by the neighbours on level 1. In case of failure or unsatisfactory response the request is transmitted to level 2 of the architecture where it will be processed by a super-peer covering the area. The latter can also route the request to another cluster in case of failure before routing as a last option to level 3 of the architecture i.e. Sparql Online Endpoints.

To form their views, peers rely on two proximity functions $S(P, Q)$ and $G(P, R)$ which, given peers P, Q and R, give

respectively a metric of the semantic proximity between P and Q and a metric of the geographical proximity between P and R.

V. Algorithms and processes for each level

In the rest of this article, we focus on the construction and evaluation of level 1 and 2 of the MoRAI model. Level 3 is relying on classical SPARQL protocol connections and will not be covered here. In MoRAI, member peers' profiles are initialized according to the three views they hold. On each level and at each cycle, a predefined process is executed.

Algorithm 1 GeoRank Algorithm

Require: P a peer profile, Candidates a set of peers profiles of size c
Ensure: G a view of size s_g
  $G \leftarrow \emptyset$
  for $i \leftarrow 1$ to c **do**
    $distance \leftarrow ComputeDist(P, Candidates(i))$
    $SetScore(P, Candidates(i), distance, score_{P,C_i})$
  end for
  $bestCandidates \leftarrow RankCandidatesByScore(score_{P,C_1}, score_{P,C_2}....score_{P,C_c})$;
  for $j \leftarrow 1$ to s_g **do**
    $G \leftarrow G \cup \{bestCandidates(j)\}$
  end for

  return G

  $SetScore(C_1 \in Candidates, C_2 \in Candidates, distance\ d, score_{C_1,C_2})$
  $score \leftarrow 0$
  if $d > 0$ and $d <= 15000$ **then**
    $score \leftarrow score + 15$
  else if $d > 15000$ and $d <= 30000$ **then**
    $score \leftarrow score + 10$
  else if $d > 30000$ and $d <= 45000$ **then**
    $score \leftarrow score + 5$
  else if $d > 45000$ **then**
    $score \leftarrow score + 0$
  end if
  $score_{C_1,C_2} \leftarrow score$

A. Level 1

To build the RPS and SON overlays we reproduced and extended the approaches of [1], [3]. We added a method to build the GON and modified the execution phase of these protocols to allow each peer to query a super-peer.

- `Random Peer Sampling (RPS) overlay` is initialized and maintained with the Cyclon [7] protocol. To update the RPS, a peer periodically performs a shuffling i.e., selects the oldest peer from its view and they exchange parts of their views. This view is used for the bootstrap and the maintenance of the semantic and the geographic overlays. The random choice also avoids cases of peer isolation.
- `Semantic Overlay Network (SON):` is built on top of the RPS using the ranking algorithm defined in Snob [1]. It uses triple patterns containment to define similarities among profiles and assigns weights to the containment to rank neighbours. SON clusters browsers according to their profile. The maintenance of s_s-best neighbours is performed at RPS exchange time applying a similarity metric to the queries. To compare peer profiles, we test triple pattern containment by searching for substitution of variables in the triple patterns.
- `Geographic Overlay Network (GON):` the GON is also built on top of the RPS. To calculate distance between peers we use the geographical coordinates embedded in each profile. At the execution of the shuffling each peer also updates its geographical view. This update is done by replacing a neighbour in the GON with the oldest one from the RPS. The $GeoRank$ algorithm then allows to order all these neighbors and to assign a score to each distance between neighbors. The $RankCandidatesByScore$ function returns Canditates with peers ranked by their computed score $score_{P,C_i}$. The $ComputeDist$ function returns the Euclidean distance between two peer profiles. The $SetScore$ function assigns scores to distances.

In Level 1 the predefined process consists of three sub-processes executed in each cycle by each participating peer: Periodic Shuffling, DataStore Update and Mobility Execution.

Periodic Shuffling: in this sub-process the three RPS, SON and GON views held by the peer are updated. For each view, the update is started with a choice of a neighbor. Suppose that a peer P participating in the architecture is initialized with a view $V = \{V_1, V_2, ...V_{s_v}\}$ and V_i ($1 \leq i \leq s_v$ with s_v the corresponding view size) a neighbor of P belonging to V. The update of V starts with a choice of a V_i. For the RPS this choice is made in a random way, while it is based on profiles similarity for the SON case, and for the GON it is based on the geographical closeness. Finally a data exchange is started with V_i in case it is active, otherwise V_i is removed from V.

DataStore Update: this sub-process updates the local RDF triplestore of each peer. The update starts by sending all local SPARQL templates to all neighbors (RPS, SON and GON). Each neighbor receiving these SPARQL queries executes them locally and returns responses to the initiating peer. All RDF results are then stored in the initiator local RDF triplestore.

Mobility Execution: this sub-process permits to consider peer mobility. During its execution, peer's geographical coordinates may be updated to a new position. This makes it possible to prepare the next GON shuffling. Considering peers' mobility, displacement induces a coordinate update which then implies a GON update. This provides opportunity to take advantage of new neighbors to acquire new data responding to local requests.

B. Level 2

This level consists of super-peers covering the architecture's target zone. By analogy to our scenario in Section II, the zone could be the city of Saint-Louis. They are set up according to the structure of a structured peer-to-peer architecture allowing requests to be directly routed between super-peers.

Super-peers are physically supported by public buildings that can ensure stable Internet connectivity and energy autonomy. We make this hypothesis because in reality, in large African cities in general, Saint Louis in particular, public structures such as the prefecture, the governance, the art museum, etc., are provided with electrical generators to compensate probable power outages. These structures are also connected to Internet by special lines to ensure permanent service stability and continuity. Installing WiFi terminals on these structures allow us to cover the different areas targeted by MoRAI and to offer local connectivity to users.

In the initial state, the data graph is distributed over all super-peers according to the different thematic. This means that on each cluster of super-peers is applied a total replication [2] of a part of the graph (a sub-graph) representing a given thematic. The super-peers belonging to the same cluster thus locally store (in local triplestore) the same sub-graph. This allows the system to reduce data exchange in the cluster. On the other hand, extra-cluster links are established between different cluster of super-peers. This allows the routing operation between clusters to recover the totality of possible answers for a given request. On level 2, apart from the initialization phase, two sub-processes are executed by super-peers for each new request received from level 1 peers: Execution/Routing and Compilation/Restitution sub-processes.

Execution/Routing: each peer of level 1 chooses, at each cycle, its super-peer contact according to the zone where it is located and its distance from the super-peer. The nearest super-peer according to distance is chosen as contact and the request is sent. The contact super-peer then locally processes the request and routes it to neighboring super-peers from different clusters. Routing operation is carried out through the existing extra-cluster links.

Compilation/Restitution: each neighboring super-peer receiving a request through an extra-cluster link performs a local processing before sending its answers back to the initiating super-peer. Once all responses are received from routing operation, the contact super-peer compiles all these responses, updates the local datastore (the local sub-graph) and forwards datas to level 1 peer initiating the request.

VI. Experimental Evaluation Through Simulation

To evaluate MoRAI, we experimented the architecture in simulated environments configured from the characteristics of our motivating scenario. Our objective is to have a simulation environment to reproduce, compare and evaluate different architectures from the state of art and from our own design.

We compared the performance of MoRAI to the state of the art Snob model [1] with the all usual metrics: request completion rate, network load (number of messages on the network) and resistance to members' failures. Our general hypothesis is that the introduction of super-peers and GON will lead to a better query completion rate than Snob without a significant increase (or acceptable increase) of network load. This will allow us to maintain the limit on the number of messages which is one of the strong points of Snob ($number\_of\_messages < (all\_overlays\_size * |query|)$. We also hope to maintain an optimal completion rate when MoRAI faces failure scenarios. We are also looking at the number of requests received and processed by Level 2 super-peers to determine this level's impact on the completion rate.

A. Experimental Setup on top of Peersim

We build our experimentation environment with the Peersim [8] tool. We parameterize the environment with 196 contributor peers and 4 super-peers. We use real datasets used in the evaluation of Snob: Diseasome and Linked Movies Database (Linked-MDB). These datasets represent two points of interest for the participating peers. We add to this dataset a manually created dataset (GeoCoord) containing geographic coordinates. These coordinates are representative of the initial positions of the different participating peers. They are chosen so that the distances between peers range from 1 to 50000 metres. We choose these parameters to get closer to our example scenario of Saint Louis festival. In this scenario the 196 peer contributors would represent people attending the festival. As we do not have real and reliable data on the festival's topics, we are using data related to Disesomes and LinkedMDB, which will represent the two major topics of the festival, namely concerts and cultural activities. This allows us to perform the simulation with two different points of interest. Our 4 super-pairs organized in two clusters of two super-pairs represent in the scenario the super-pairs installed in public buildings with reliable internet access and high storage capacity. The two clusters would be located, for example, at the city's Stadium where concerts will take place and at the Arts Museum where cultural activities will be held. We have chosen a maximum distance of 50000 meters between participants to contain all participants within a limited perimeter. For reproducibility the complete dataset is available online[1].

Each peer is thus initialized with a set of randomly selected triples in Diseasome and LinkedMDB. Each peer is also initialized with random geographic coordinates from GeoCoord. The super-peers are parameterized to form two clusters of two super-peers. This means that the graph is split into two parts (sub-graph), each part is stored by a cluster. The super-peers belonging to the same cluster therefore hold the same sub-graph. We also note that we set the system such that the two sub-graphs are representative of the two topics considered (Diseasome and LinkedMDB), which also implies that the clusters are thematic.

We reuse the generated query set for Snob [1] evaluation. These are 100 queries for LinkedMDB and 96 queries for Diseasome. To simulate architecture points of interest, we

[1] https://github.com/MmoooGit/MoRAI

associate geographic coordinates to each request. This allows to consider the position of the request holder peer in relation to the point of interest. Peers are initialized randomly with zero or two queries. To calculate the completion rate on each execution round, we also associate each query with a cardinality representing the size of the set constituting the data graph triples that match the query.

To simulate the mobility of the participating peers, on each execution round, an update of the geographical coordinates is performed with a random choice between three pairs of coordinates which are: the actual coordinates (the peer is motionless since the last round), the coordinates carried by one of the stored requests (the peer moves towards one of its points of interest), a pair of random coordinates (the peer moves towards an arbitrary place). The simulation has two phases: the first one with a normal execution of each cycle for all participating peers; the second one with an execution disrupted by crash scenarios to observe the impact of peer failures on the different metrics.

Normal Execution: A total of 196 participant peers are initialized. 98 of them hold 2 requests each and the remaining 98 hold 0 requests. That is a total of 196 requests loaded for the simulations. Given that the Snob model (RPS+SON) has already made comparisons with the RPS model and has proven its efficiency, we compare the Snob model with the RPS+SON+GON and RPS+SON+GON+SP models.

- `RPS+SON`: This is Snob model simulated in our environment for reproducibility and comparison. We modify the initial model (RPS+SON) by initializing the peers with 2 queries (2Q) instead of 1 for the initial model.
- `RPS+SON+GON`: idem and we add a geographical view (GON) to take into account the location of the peers during execution.
- `RPS+SON+GON+SP`: idem and we add the level 2 of the architecture represented by the super-peers (SP).

Crash Execution: we are interested here in peer failures since in our context they are the most volatile compared to super-peers. Our crash simulations will focus only on peer failures. In these scenarios we choose a series of execution rounds during which we drop a specific number of peers. In order to get an accurate view of the impact of the requests, these peers are chosen from those that hold requests.

B. Experimental Results

Completion rate: we notice a clear improvement of the completion of the RPS+SON+GON model compared to the RPS+SON model (see Fig. 2 and Fig. 4). This can be mainly explained by the fact that, at each execution round, the GON brings new data to the initiator peer. Based on the hypothesis that geographic neighbours can be an important reserve of responses, choosing to query them increases the number of neighbours potentially holding favourable query responses. The completion rate gains a much greater improvement when we add the super-peers to the architecture. Indeed we notice on the curve that the completion rate reaches 100% after 12 rounds on average against 25 rounds for the RPS+SON+GON model and 45 rounds for the RPS+SON model (see Fig. 2). This is explained by the fact that the data graph is fully replicated by the two clusters of super-peers and that the inter-cluster links allow super-peers from different clusters to exchange. This allows each participating peer to receive the maximum possible response to its requests. In order to have an accurate estimation of the impact of super-peers, we have configured the simulation environment so that each participating peer has only one access to the super-peers. We restricted this access to super-peers to the first round to show their impact as architecture starter. From this single access we can see the impact of super-pairs on request completion rates in the next rounds.

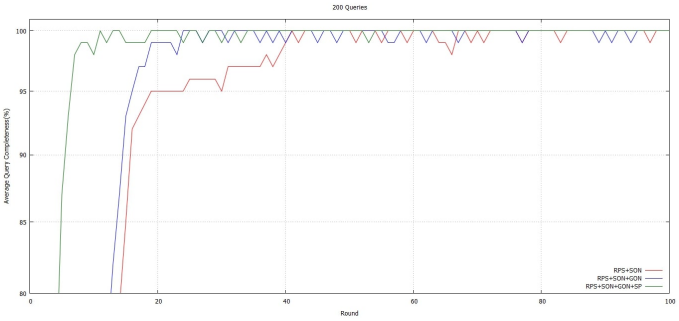

Fig. 2. Average query completeness by round for RPS+SON, RPS+SON+GON and RPS+SON+GON+SP, with 196 peers, 196 queries, 2 clusters of 2 super-peer. RPS, SON and GON sizes: respectively 10 (5 for shuffling size), 5 and 5.

Network load: the load observed for the RPS+SON+GON and RPS+SON+GON+SP models is 33% higher than the load of the RPS+SON model (see Fig. 3). This is explained by the fact that GON involves additional exchanges on the network. Indeed, for each participant, in addition to the exchanges generated during the shuffling at the RPS and SON level, the GON is called upon for local processing of requests during the periodic shuffling. This load increase is, however, compensated by the completion clearly visible on the curve. The two models RPS+SON+GON and RPS+SON+GON+SP generate approximately equal amounts of messages. This is due to the fact that exchanges between layer 1 and layer 2 generate few messages since we have limited them to a maximum of one message per peer. This translates into a maximum increase of 96 messages over the 100 execution rounds, or 4,8% of the load of the RPS+SON+GON model.

Although this load increase is not negligible, it is nevertheless acceptable in our scenario because if we consider that the limit of the total number of messages is defined by $(R_s + S_s + G_s) * |queries|$ i.e (5+5+5)*196= 2940 messages, we note on the curve that the maximum of the number of messages is 2003 messages thus lower than the limit. We can suppose this limit as only one pattern is addressed to each neighbor. Also the load induced by the exchanges with super-peers is acceptable since we have a limited number of super-peers and the objective is that they be requested as less as possible.

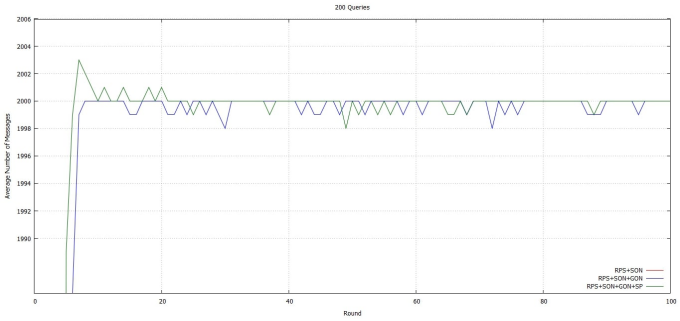

Fig. 3. Average number of message by round for RPS+SON, RPS+SON+GON and RPS+SON+GON+SP with 196 peers, 196 queries executed, 2 clusters of 2 super-peer. RPS, SON and GON sizes: respectively 10 (5 for shuffling size), 5 and 5.

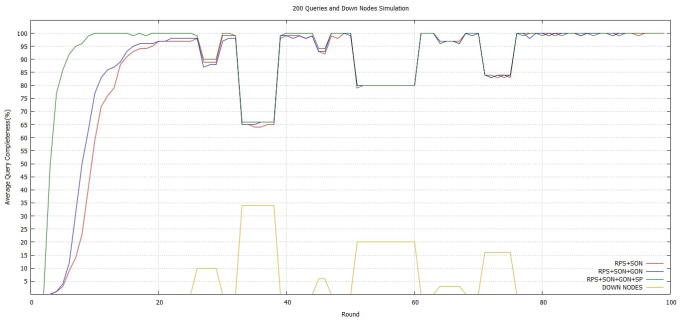

Fig. 4. Average query completeness by round with crash scenario. DOWN NODES yellow curve at the bottom shows the number of nodes down per round. RPS+SON, RPS+SON+GON and RPS+SON+GON+SP are tested with 196 nodes, 196 queries executed, 2 clusters of 2 super-peer. RPS, SON and GON sizes: respectively 10 (5 for shuffling size), 5 and 5.

Failure Resistance: we observe a global resistance of the system to the provoked failures. On the three models RPS+SON, RPS+SON+GON, RPS+SON+GON+SP the curve (see Fig. 4) shows a fall of the completion approximately equivalent. Indeed the three models show similar reactions to the crash scenario. The curves reach the same completion rates within the predefined crash intervals. This is precisely explained by the fact that the completion rate is calculated as a function of the number of living (up) peers in the system and the local completion rate of each of these peers. The local completion rate of a peer is obtained as a function of the total number of potential responses available on the entire graph to answer its local requests. Since the number of peers falling on the different crash intervals are equivalent across all three models, completion follows the same equivalence. However, we notice on the curve a slight completion difference at the intervals [25-30] and [33-38]. This is caused by the overall completion level of the RPS+SON+GON+SP model which, for these intervals ([25-30] and [33-38]) had already reached 100% of completion against (respectively) 98% and 99% for the two others.

VII. Conclusion

In this paper, we proposed and evaluated MoRAI (Mobile Read Access in Intermittent internet connectivity), a distributed peer-to-peer architecture organized in three levels dedicated to data exchanges by mobile contributors. We designed MoRAI to support the local exchange of RDF data. Regarding the requests completion rate and the system's resistance to crash scenarios, our experimental evaluation results show the relevance of considering geographical positions during data exchanges and of integrating RDF graph replication to ensure data availability on the architecture.

Our next step is to deploy and evaluate this approach in an application allowing data sharing during a real event. We also plan to design a more advanced version of MoRAI which will integrate the write access of peers so they can collaboratively modify the RDF graph they host. This will allow us to consider in the future the generalization to other scenarios such as distributed semantic tagging in museums or tourism information sharing.

References

[1] AGrall, A., Skaf-Molli, H., Molli, P.: SPARQL query execution in networks of web browsers, (2018)

[2] Spaho, E., Barolli, L., Xhafa, F.: Data replication strategies in P2P systems: A survey. In: 2014 17th International Conference on Network-Based Information Systems, pp. 302-309, IEEE, 2014

[3] Folz, P., Skaf-Molli, H., Molli, P.,: CyCLaDEs: a decentralized cache for Linked Data Fragments. In: ESWC: Extended Semantic Web Conference, 2016

[4] Hong Kong Protests Propel FireChat Phone-to-Phone App, https://www.nytimes.com/2014/10/06/technology/hong-kong-protests-propel-a-phone-to-phone-app-.html. Last accessed 22 May 2020

[5] Sambra, Andrei V., Mansour, E., Hawke, S., Zereba, M., Greco, N., Ghanem, A., Zagidulin, D., Aboulnaga, A., Berners-Lee, T.: Solid: a platform for decentralized social applications based on linked data. Technical report, MIT CSAIL & Qatar Computing Research Institute, 2016

[6] Hernández, Antonio G., GARCıA, MNM.: A JavaScript RDF store and application library for linked data client applications. In: Devtracks of the, WWW2012, conference. Lyon, France, 2012

[7] Voulgaris, S., Gavidia, D., Van Steen, M.: Cyclon: Inexpensive membership management for unstructured p2p overlays. Journal of Network and systems Management, **13**(2), 197–217, Springer, 2005

[8] Montresor, A., Jelasity, M.: PeerSim: A scalable P2P simulator. In: 2009 IEEE Ninth International Conference on Peer-to-Peer Computing. pp. 99–100, IEEE, 2009

[9] E. Rosas, F. Garay, N. Hidalgo and O. Marin, : Mobility-aware DTN protocols for post-disaster scenarios, 2015 2nd International Conference on Information and Communication Technologies for Disaster Management (ICT-DM), Rennes, 2015, pp. 193-199, doi: 10.1109/ICT-DM.2015.7402038.