- Many decentralized web initiatives
- Give (or re-give depending on perspective) control/sovereignty over data [1]
    - Where they store data [1]
    - Who has access to it [1]
- Is it enough?
    - Divergeance of interest or representation (topics, design choices, languages) [4]
    - Utilization of users as unpaid source of revue [5] even when data is anonymized [2] [4]
    - Contribution to centralization of wealth [3] [4] [5]
- We could consider giving the user a more democratic control of their social application
    - A structural way to do it is by making them give the computational power for the functioning of the applications

Introducing Collaborative Link Traversal Query Processing (CLTQP) in the Context of Structured Decentralized Environments

Bryan-Elliott Tam

Universiteit Gent

Agenda

1. Structured decentralized linked data environment and social applications
2. What we are found in the literature
3. Problem of completeness of queries and execution time
4. Research questions and hypotheses
5. Planned Solution for Search for domain division and collaborative caching
6. Limitation and what else can be done
7. Conclusion

Structured decentralized linked data environment and social application

A structured decentralized linked data environment is a decentralized environment . Where we can consider “additional structural properties [along with the linked data principals] that allow us to make additional assumptions about data, documents, and their organization during query execution.” [21]

Examples of specification providing structure

- Linked Data Platform (LDP)
- Type Indexes
- WebID

Structured decentralized linked data environment and **social applications**

- Application driven by the interaction between users
- Core functionalities
  - Requesting information
  - Discovering information

What we are found in the literature

- Link Traversal Query Processing (LTQP)
- Collaborative SPARQL Querying
- P2P caching in the context of the web

Link Traversal Query Processing (LTQP)

- Recursively looking up URIs to explore surrounding information, starting from a set of URIs called seed URIs [7]
- Predefined lookup policies with no input from the client [7] [9]
- Difficulties; open-endedness of the web and query planning [8]

Collaborative SPARQL Querying

Using multiple agents to facilitate querying by diminishing the computation load of the execution or the discovery of data sources. [10]

- Rotating data sources in a P2P network; Snob [11]
  - Partial view of the network; random and profile based
  - Results share after the rotation
  - Possible to enhace completness without visiting each data sources

Collaborative SPARQL Querying

- Division of the search space in comunity: ColChain [12]
  - Creation of deliberate comunities
  - Querying own data source then the ones from known communities.
  - Collaboration in the division of the search space based in socially close information
- Leveraging the social links between data sources [13] [14]

P2P caching in the context of the web

- Mapping URLs in a distributed hash table (DHT): Squirrel [15]
- Flower-CDN add a mechanism to consider the localicality of the node
- Using an unstructured network with partial views; random and profiles based ("behavioral locality"): Behave [17]
- CyCLaDEs adapt the concept of Behave to SPARQL queries of RDF documents [18]

What is the scientific problem we want to solve?

Problem of completeness of queries and execution time

- Define the domain as a subset of a graph containing all the triples available in a decentralized structured environment
- With more engines, more exploration of the domain
    - Search of non overlapping section
    - Sharing of results
- Structural biais of results
    - Sensibility of initial condition
    - Popularity biais

Problem of completeness of queries and **execution time**

We propose to use caching

- Multiple similar, complementary or identical queries
- In LTQP caching can improve query completness, but can increast query execution time [22]
    - Has not been evaluated in structured environment
- In the literature does not consider LTQP particularities: long execution time, multiple sources, difficulties to attain completeness

Research questions and Hypotheses

1. Can we achieve better query result completeness and lower global query execution time in the context of LTQP by making multiple SPARQL query engines collaborate?
2. Does the CPU usages and the number of HTTP request for each engine diminish with the increase of engine collaborating?
3. How can we prevent multiple query engines from repeating identical calculations over the same data sources?
4. How can we reduce query execution time using P2P caching in the context of CLTQP?

Research questions and **Hypotheses**

1. There is an inverse correlation between the number of engines collaborating and the execution time, and a direct correlation with the number of data sources explored.
2. It is possible to partition the search space in the context of CLTQP, in a way that the query processing time of overlapping data sources is less than the time to process distinct data sources.
3. It is possible to index a P2P cache and create a procedure for its usage in the context of CLTQP so that its lookup time is faster than the execution of the query.

What are our potential solutions?

Search domain division and collaborative caching

1. Collect the seed URLs and divide them between the query engines
   - Minimal comunication between the engines
   - Do not consider the overlaps
   - Potential loss of accuracy if the solutions are divided between multiple engines
2. Set the reachability criteria of each engine so that they cannot or are less likely to have overlapping search field
   - The criteria has to be changed depending on the type of query
3. Use a global link queue and solution map shared between all the query engines
   - Necessitate more comunication between the engines

Metric evaluated

- Result accuracy
- Query execution time
- The ratio of the execution time and the communication time between the engines
- Ability to access isolated documents
- Overlapping of the search space
- Query result arrival times

Search domain division and **collaborative caching**

- Caching of URLs contributing to the query
- intermediary joint results
- Three interpretations
  - Directory of (partial) results
  - Checkpoint for longer execution
  - Map of data source to explore

1. Unstructured network where peers are clustered based on their behavior
2. Distributed Hash Table to find the cached element

Metric evaluated

- Cache access time
- Cache miss and cache hit rates
- Execution time reduction

Limitation and what else can be done

- Mechanism for detecting similar queries; eg: Query containment [27]
- Fairness/"social contract" for sharing of computational power and cache
- Measurement to know if it lowers the cost of social applications (it's hard to compare because it has plus value that is hard to quantify)

Conclusion

- Building a Collaborative Link Traversal Query Processing in the Context of Structured Decentralized Environments
- More complete queries by making more engines collaborate
- Faster queries by sharing cache information
- Aim to make the creation of social applications more democratic

References

1. Verstraete, M., Verbrugge, S., Colle, D.: Solid: Enabler of decentralized, digital platforms ecosystems. In: ITS (2022).
2. Terranova, T.: Free Labor: Producing Culture for the Digital Economy. Social Text. (2000).
3. Curran, J.: The internet of dreams Reinterpreting the internet. In: Misunderstanding the Internet (2016).
4. Sevignani, S.: The commodification of privacy on the Internet. Science and Public Policy. (2013).

References

1. Mechant, P., De Wolf, R., Van Compernolle, M., Joris, G., Evens, T., De Marez, L.: Saving the web by decentralizing data networks? A socio-technical reflection on the promise of decentralization and personal data stores. In: 2021 14th CMI International Conference (2021).
2. Smets, A., Michiels, L., Bogers, T., Björneborn, L.: Serendipity in Recommender Systems Beyond the Algorithm: a Feature Repository and Experimental Design. In: IntRS@RecSys (2022).

References

1. Hartig, O., Özsu, M.T.: Walking Without a Map: Ranking-Based Traversal for Querying Linked Data. In: ISWC
2. Hartig, O.: Linked Data Query Processing Based on Link Traversal. In: Linked Data Management
3. Hartig, O., Freytag, J.-C.: Foundations of Traversal Based Query Execution over Linked Data. In: Conference on Hypertext and Social Media