Distributed Search with Rendezvous Search Systems

Christof Leng

Technische Universität Darmstadt, Germany
cleng@dvs.tu-darmstadt.de

I. Introduction

The ever-growing amount of data made available through the Internet requires powerful and highly scalable search systems. Not only the mostly unstructured data but also the increasing user expectations demand for sophisticated and flexible search primitives like keyword search or XPath. Academic research on peer-to-peer networks has been mostly focused on how to build such advanced primitives on top of the key-value-lookup primitive offered by Distributed Hash Tables (DHTs) [2], [10]. While in most cases possible, this approach often requires expertise in both distributed systems and information retrieval beyond the typical application developer.

Rendezvous search systems take a completely different approach by separating the matching of queries and data from the network communication. This not only allows to use the same rendezvous search system unchanged for many different search primitives, but also to easily integrate existing non-distributed implementations of search primitives. The application developer provides the rendezvous search system a data storage backend and a match function to determine matching data from the backend for an incoming query and the rendezvous search system takes care of the distribution of data and queries in the distributed environment. In my talk I want to give a short overview of the current state of the art of rendezvous search systems.

The name rendezvous search system refers to the guarantee that a query meets a copy of every data item in the system eventually. How to fulfill this guarantee is up to the rendezvous search system and depends on the target environment, but typically requires $O(\sqrt{n})$ copies of queries and data respectively.

II. Data Center

In very static environments like data centers Google's approach [1] with a grid-like layout of nodes is very efficient and easy to implement. A document is replicated on each node in a random row and a query is evaluated against the data of all nodes in a random column. This approach is inspired by the grid protocol used in quorum systems.

For unpredictable workloads ROAR [9] uses a ring instead of a grid. By putting data on ring sections and querying equidistant nodes on the ring, the rendezvous guarantee is fulfilled. The ring structure not only allows the dynamic change of system size but also to dynamically adjust the number of data or query replicas to optimize system traffic or response times under changing workloads.

III. Structured P2P

Bit Zipper [11] is an add-on for DHTs which enables rendezvous search mechanisms on an existing DHT infrastructure. Data and queries are distributed through a fractal tree. The rendezvous guarantee is kept if the DHT routing is consistent.

Deetoo [3] takes a slightly different approach and maps the grid on two DHT rings, one for the columns and one for the rows. Every node in Deetoo participates in both rings.

IV. Unstructured P2P

For even rougher environments, unstructured peer-to-peer overlays can provide the necessary resilience. The unstructured rendezvous search systems use tunable probabilistic success guarantees, which are proven with combination of probability theory and graph theory.

The rendezvous search system from Ferreira et al. [4] uses random walks to distribute both queries and data on an expander graph topology. BubbleStorm [12] instead uses a form of constrained flooding and a random graph topology to improve performance and resilience. Hautakorpi and Schultz [5] use an unstructured random walk approach on a DHT topology and show how to incrementally deploy their system on an existing DHT system.

SplitQuest [8] combines ideas from structured and unstructured rendezvous search systems by using a key-based routing scheme on an unstructured topology. The first simulation results are promising, but the system lacks the thorough theoretical foundation of its siblings.

In unstructured rendezvous search systems it is non-trivial to sustain the replica count of data and to execute consistent updates. The replication mechanism [7] provides eventual consistency and replica maintenance.

V. Conclusion and Outlook

In summary, the rendezvous concept offers a wide range of powerful and flexible solutions for future Internet search systems. Even implementing a query language as sophisticated as SQL on top of rendezvous search systems is feasible [6].

The rendezvous approach offers some similarities with key-value stores in the area of cloud computing. Both use a combination of partitioning and replication to improve scalability, response times, and robustness, but rendezvous search systems feature much more self-adaptivity to cope with unstable and dynamic environments. It would be interesting to transfer some of those sophisticated mechanisms from rendezvous search to cloud storage.

References

[1] L. A. Barroso, J. Dean, and U. Hölzle. Web search for a planet: The google cluster architecture. *IEEE Micro*, 23(2):22–28, 2003.

[2] A. Bonifati, U. Matrangolo, A. Cuzzocrea, and M. Jain. Xpath lookup queries in p2p networks. In *Proceedings of WIDM'04*, pages 48–55. ACM Press, 2004.

[3] T. W. Choi and P. O. Boykin. Deetoo: Scalable unstructured search built on a structured overlay. In *HOTP2P - International Workshop on Hot Topics in Peer-to-Peer Systems*, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[4] R. A. Ferreira, M. K. Ramanathan, A. Awan, A. Grama, and S. Jagannathan. Search with Probabilistic Guarantees in Unstructured Peer-to-Peer Networks. In *Proceedings of P2P'05*, pages 165–172, Washington, DC, USA, 2005. IEEE Computer Society.

[5] J. Hautakorpi and G. Schultz. A feasibility study of an arbitrary search in structured peer-to-peer networks. In *Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–8. IEEE, Aug. 2010.

[6] C. Leng and W. W. Terpstra. Distributed sql queries with bubblestorm. In K. Sachs, I. Petrov, and P. Guerrero, editors, *From Active Data Management to Event-Based Systems and More*, volume 6462 of *Lecture Notes in Computer Science*, pages 230–241. Springer, nov 2010.

[7] C. Leng, W. W. Terpstra, B. Kemme, W. Stannat, and A. P. Buchmann. Maintaining replicas in unstructured P2P systems. In *Proceedings of CoNEXT'08)*. ACM, Dec. 2008.

[8] P. Lopes and R. A. Ferreira. Splitquest: controlled and exhaustive search in peer-to-peer networks. In *Proceedings of the 9th international conference on Peer-to-peer systems*, IPTPS'10, pages 6–6, Berkeley, CA, USA, 2010. USENIX Association.

[9] C. Raiciu, F. Huici, M. Handley, and D. S. Rosenblum. Roar: increasing the flexibility and performance of distributed search. In *Proceedings of SIGCOMM'09*, pages 291–302, New York, NY, USA, 2009. ACM.

[10] P. Reynolds and A. Vahdat. Efficient peer-to-peer keyword searching. In *Proceedings of Middleware'03*, pages 21–40, June 2003.

[11] W. W. Terpstra, S. Behnel, L. Fiege, J. Kangasharju, and A. Buchmann. Bit Zipper Rendezvous—Optimal Data Placement for General P2P Queries. In *EDBT'04 Workshop on Peer-to-Peer Computing and DataBases*, March 2004.

[12] W. W. Terpstra, J. Kangasharju, C. Leng, and A. P. Buchmann. Bubblestorm: resilient, probabilistic, and exhaustive peer-to-peer search. In *Proceedings of SIGCOMM'07*, pages 49–60, New York, NY, USA, 2007. ACM.