Tackling inter-service RDF communication bottlenecks in the Nanopublication network with Jelly

Piotr Sowiński (NeverBlink), Tobias Kuhn (Knowledge Pixels), Karolina Bogacka (NeverBlink)

Initial situation

Knowledge Pixels (a Swiss company) is leading the initiative of establishing a global Nanopublication Network[1], a decentralized system for publishing, exchanging, and querying scientific knowledge. The Network is built on RDF and Semantic Web technologies, serving researchers, academic publishers, and GLAM institutions. Knowledge Pixels aims to scale this infrastructure to an unprecedented scale, creating a global, collaborative and decentralized knowledge graph. The Network is designed to be scalable, relying on distributed services – Nanopub Registry and Nanopub Query – that respectively handle Nanopublication storage and querying. The services must frequently communicate with each other to replicate the Network's state (e.g., new Nanopublications being published). This communication can quickly become one of the main bottlenecks for the Network, as RDF serializations are generally slow and inefficient. Additionally, each Nanopublication is a separate RDF document that needs to be fetched individually, which forces repeated HTTPS requests, further slowing down communication. Hence, a better inter-service RDF communication solution was required.

Approach and solution

The solution was implemented by NeverBlink (a Polish company) using the open-source Jelly-RDF serialization format. Jelly is based on Protocol Buffers, a fast and robust binary serialization framework. It employs an efficient streaming compression algorithm and has an open-source implementation for Apache Jena, Eclipse RDF4J, and Titanium. Being a binary format with a fast compression scheme, it achieves 2.5x faster serialization and 5x faster deserialization speeds than any other format in Jena, on a mix of 13 datasets from RiverBench. It also supports streams of multiple RDF graphs or datasets, a feature particularly well-suited for this use case.

We implemented a Jelly-based communication protocol in Nanopub Registry and Query services, using the readily available RDF4J integration, replacing the existing implementation based on W3C TriG. The Registry now exposes an interface to retrieve entire batches of Nanopublications, as a single, highly efficient data stream. This is used by other Registry instances for replication, and the Query service for updating the user-facing SPARQL endpoints. Jelly is here a key enabler for the Nanopublication Network, making it possible to build a structured, global scientific discourse ecosystem, moving beyond unstructured text and dispersed data sources.

Success criteria and the benefit of the solution

The most crucial factors for success were Jelly's extraordinary performance, stability (full coverage of RDF 1.1 verified through extensive testing), and the ready integration with RDF4J, which allowed for a quick implementation and deployment. We have verified in a full deployment that Jelly reduced the time to retrieve 60000 Nanopublications (1.4M quads) to less than 4 seconds (including database reads, Internet communication and RDF parsing) from over 60 minutes. This effectively made inter-service communication a non-issue (<1% of application time) and allowed the Network to reach the scale needed for a global scientific knowledge graph.

Prospects and recommendations

We are currently scaling up the production infrastructure and implementing a very large-scale test scenario, intended to demonstrate a decentralized, many-billion-quad scientific knowledge graph. In such a setting, sharding and SPARQL federation can be employed to scale the system, however then, the communication becomes a major concern. Given the excellent performance of Jelly so far, we expect this to be successful. Similar problems as the one presented here arise often in client-server and microservice architectures. Therefore, we expect Jelly to also be able to benefit other networked Semantic Web applications, improving efficiency, lowering costs, and making for more competitive products. We are now exploring applications in database replication, RDF change capture, RDF visualization, and accessing cloud-based triple stores.

---

[1] Kuhn, T., Taelman, R., Emonet, V., Antonatos, H., Soiland-Reyes, S., & Dumontier, M. (2021). Semantic micro-contributions with decentralized nanopublication services. *PeerJ Computer Science*, *7*, e387. https://doi.org/10.7717/peerj-cs.387