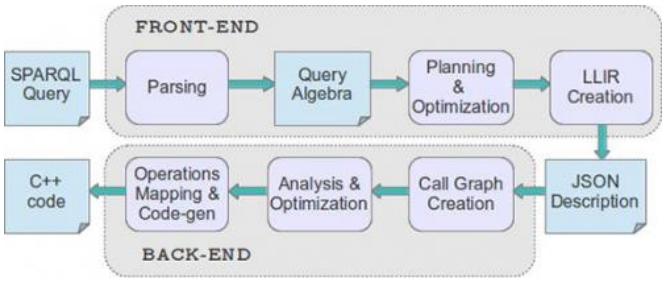
Cooperative software framework helps tame "too big" data

23 March 2015



Automatic code generation flow in GEMS' SPARQL-to-C++ compiler. The front end processes the SPARQL query and creates of a low-level internal representation (LLIR), which is exposed to the back end. The back end is responsible for translating the LLIR into optimized C++ code (JSON: JavaScript Object Notation).

As data sets grow increasingly large and heterogeneous (or, "too Big"), their value diminishes if they cannot be mined with precision and purpose. The Semantic Web adds meaning to information on the Web by promoting common data formats that can be more readily located, shared, and used. Often, the RDF model, specifications originating from the World Wide Web Consortium, is used to link varied data, making the information more viable for query and analysis. In earlier research, GEMS was shown to manage data volume challenges associated with employing a graph-based data model. This time, data mining through graph methods using the GEMS framework on currently available computing components, or commodity clusters, affords more efficient use of space and added performance by exploiting graph parallelism.

Furthering work involving the Graph Engine for Multithreaded Systems, or GEMS, a multilayer software framework for querying graph databases developed at Pacific Northwest National Laboratory, scientists from PNNL and NVIDIA Research used GEMS to customize commodity, distributed-memory high-performance computing (HPC) clusters and apply graph algorithms to large-scale data sets on clusters. By incorporating GEMS, HPC query solutions, such as parallel processing, are exploited and results are more predictable. Moreover, GEMS translates SPARQL queries, a Resource Description Framework (RDF) query language, to C++, a general, cross-platform programming language, more efficiently to optimize HPC-based graph-matching methods. In their comparison with alternative approaches, GEMS provided noticeable speedups, particularly with larger data sets.

This work is featured as part of the March 2015 special issue of the Institute of Electrical and Electronics Engineers Computer Society's flagship publication, Computer, devoted to efforts surrounding Big Data management.

Different from other RDF engines, which resort to more conventional relational databases approaches, GEMS mostly employs graph methods to process SPARQL queries. The core syntax of SPARQL is a conjunctive set of triple patterns, called the "Basic Graph Pattern," which represents subgraphs to match against the RDF data. In GEMS, SPARQL queries are modeled as graph homomorphism routines, enriched with solution modifiers (e.g., sorting, aggregation) to support the features offered by SPARQL. These methods are automatically implemented in C++ by mapping the basic query operations to highly parallel functions selected from GEMS' Semantic Graph Library (SGLib). To obtain efficient implementations, the translation engine includes several analysis and optimization steps, conducted on different representations of the input query, which range from its algebraic representation to dependency graphs. The optimization process identifies an optimal execution plan among several candidates, according to a cost model and a cardinality estimator. Then, it performs data-flow and call-graph analysis to improve task-level parallelism exploitation, reduce data movement, and curtail the

memory footprint of data structure. The process also identifies the particular sequence of basic operations that can be combined into more efficient complex operations.

"GEMS clearly represents a promising solution to tackle the 'too Big' challenge as it already is able to process data in the scale of 10 billion triples, which is prohibitive for most available systems," explained Vito Giovanni Castellana, a research computer scientist with the Advanced Architectures team in PNNL's High Performance Computing group and the paper's primary author. "Nevertheless, the user experience is important as much as scalability and performance. GEMS' SPARQL-to-C++ compiler allows analysts to quickly describe the queries, reducing the development time from several hours to minutes compared to the basic C++ interface. Moreover, the generated C++ code is available to the user, who can potentially tune it, instrument it, and customize it to introduce new functionalities not featured in the SPARQL standard with limited effort. This characteristic makes GEMS not only a SPARQL query engine, but a more flexible platform for Big Data analytics."

Ongoing GEMS code development will improve query performance, for example, by using statistics gleaned through data probing. Also, the researchers still are finding ways to augment C++ graph-matching operations by accounting for data characteristic sets.

More information: Castellana VG, A Morari, J Weaver, A Tumeo, D Haglin, O Villa, and J Feo. 2015. "In-Memory Graph Databases for Web-Scale Data." *Computer* 48(3):24-35. DOI: 10.1109/MC.2015.74

Provided by Pacific Northwest National Laboratory