Enabling High-Performance Computing Resources For Efficient Semantic Web Reasoning

Alexey Cheptsov¹, Matthias Assel¹, Georgina Gallizo¹,

¹ High-Performance Computing Center Stuttgart, Nobelstr. 19, 70569 Stuttgart, Germany
{cheptsov, assel, gallizo}@hlrs.de

Abstract. The paper presents a technology of high-performance computing resource’s utilization for solving complex tasks of semantic web reasoning, applied within LarKC - the Large Knowledge Collider. The technology enables utilization of a virtually unlimited resource pool provided within grid and cloud environments and facilitates obtaining high performance and scalability for large-scale reasoning tasks on the Web by applying special parallelization techniques elaborated in LarKC. The technology might also be of potential interest for many Semantic Web reasoning communities.

Keywords: Semantic Web, Reasoning, LarKC, High-Performance Computing

1 Introduction

The recent evolution of the World Wide Web towards Web 3.0 resulted in the Semantic Web – a progressive technology which potentially enables the Web content of billions of Web pages to be programmatically accessible and handled. The tremendous amount of inter-connected instance data, annotated with possibly expressive ontology models, offers several challenging scenarios for their efficient use in the large-scale reasoning engines, such as provided by Pellet, or Jena. However, performance and scalability characteristics of those systems can greatly benefit from applying parallelization techniques, which facilitate utilization of inherited algorithm parallelism when running on high-performance computing architectures, like those provided by generic clusters of workstations or dedicated supercomputers with optimized node interconnect and I/O. For the modern Semantic Web reasoning systems, e.g. OWLIM [1], several forms of parallelism can be recognized, such as inter-querying (running more than one query in parallel), intra-query (running subqueries in parallel and pipelining operators), or intra-operation (distribution single operations for concurrent execution). However, there are still many difficulties presented which arise when applying those approaches in practice for a generic high-performance resource, to which special policies for multi-user support, resource management, job scheduling etc. are generally pertained.

LarKC (the Large Knowledge Collider) [2] aims to build an experimental platform for massive distributed incomplete reasoning that will remove the abovementioned scalability barriers, in particularly by enabling integration with high-performance
computing systems. This is achieved thanks to a flexible Service-oriented Architecture, which along with a number of original solutions for obtaining web scale by Semantic Web applications ensures optimal characteristics and interoperability with the complete infrastructure.

Semantic Web components, that compose a LarKC application, are organized in form of “plug-ins” – light-weight components with self-contained functionality, which follow the “divide and conquer” strategy and can be composed in an execution workflow, or combined in an appropriate manner for an application. Thanks to the LarKC architecture design for support of distributed and remote execution, interoperability is enabled between distributed heterogeneous systems, using different platforms and technologies, permitting plug-ins and workflows composed of them to take advantage of all available resources (hardware and software) to achieve the maximum performance and scalability.

In LarKC, a virtually unlimited variety of resource configuration is enabled for plug-in deployment. Deployment options can include combinations of generic remote web servers, flexible desktop and service grids as well as cloud environment resources and others. In LarKC, this flexibility is achieved by means of special managers which enable deployment of a plug-in on a remote resource. Several resource types are supported by the remote plug-in manager’s architecture, including remote servers accessible via HTTP(FTP)/SSH(SCP) protocols as well as desktop (e.g. BOINC) and large scale service (e.g. EGEE, DEISA) grid systems accessible through the specific middleware. This is achieved thanks to adoption by LarKC of GAT (Grid Access Toolkit) technology. However, the LarKC architecture is designed in a way that allows plug-in developers to abstract from a concrete resource, adaptation to which is performed by a corresponding manager applied for a plug-in on the platform level.

For obtaining the highest performance of LarKC plug-ins deployed on a parallel architecture, several parallelization strategies have been elaborated and applied for pilot LarKC applications. In case of non-concurrent workflow scenarios, several mechanisms were studied for a “within a plug-in” parallelization, whereby support for several solutions, including shared memory parallelization (multi-threading, Open MP etc.) and distributed memory parallelization (MPI), was integrated.

The proposed poster is intended not only to present the main outcome of parallelization and distribution efforts performed within LarKC, but also initiates a discussion on more common use of high-performance resources in Semantic Web area.

References