A Space-Based Generic Pattern for Self-Initiative Load Balancing Agents

Eva Kühn and Vesna Sesum-Cavic

Vienna University of Technology, Institute for Computer Languages,
Space Based Computing Group, Argentinierstr. 4, 1040, Wien, Austria
{eva,vesna}@complang.tuwien.ac.at

Abstract. Load-Balancing is a significant problem in heterogeneous distributed systems. There exist many load balancing algorithms, however, most approaches are very problem specific oriented and a comparison is therefore complex. This paper proposes a generic architectural pattern for a load balancing framework that allows for the plugging of different load balancing algorithms, reaching from unintelligent to intelligent ones, to ease the selection of the best algorithm for a certain problem scenario. As in complex network environments there is no "one-fits-all solution", also the integration of several different algorithms shall be supported. The presented pattern assumes autonomous agents and decentralized control. It can be composed towards arbitrary network topologies, foresees exchangeable policies for load-balancing, and uses a black-board based communication mechanism to achieve high software architecture agility. The pattern has been implemented and first instantiations of it with three algorithms have been benchmarked.

Keywords: Load balancing, self-organization, autonomous agents, coordination patterns, intelligent algorithms, complex distributed systems.

1 Introduction

The rapid growth of computer systems and their complexity imposes the necessity to reconsider dynamic load balancing (LB) in order to improve the performance of the overall distributed system and to achieve the highest level of productivity. LB can be described as finding the best possible workload (re)distribution and addresses ways to transfer excessive load from busy (overloaded) nodes to idle (under-loaded) nodes. LB can take place at *local node level* allocating load to several core processors of one computer, as well as at *network level* distributing the load among different nodes. At the local level, the determining factor for load distribution is the balanced utilization of all core processors. At the network level, one must take into consideration the time needed for transferring data from a busy node to an idle node and estimate the priority of transferring, especially when the transfer itself requires more time to complete than the load assignment.

H. Aldewereld, V. Dignum, and G. Picard (Eds.): ESAW 2009, LNAI 5881, pp. 17–32, 2009.

The problem becomes even more complex in heterogeneous systems. Networks are growing constantly and therefore the intensive need of including self-* properties (self-organization, self-management, self-repairing, self-configuring, self-grouping, self-learning, self-adaptation, etc.) arise to deal with increasing complexity. We can find a wide range of LB approaches (see section 2), however, our objections address the lack of: Provisioning a general framework, autonomy and self-* properties, and arbitrary configurations. *Provisioning of a General Framework:* Existing LB approaches are very problem specifically oriented (see section 2). As there is no "one-fits-all solution, in order to find a best solution for a problem, a generalized framework is needed that allows for testing and tuning different LB algorithms for a specific problem and environment. The framework shall support easy and dynamic exchange of algorithms as well as combinations of different algorithms. The architecture shall be agile, so that neither new requirements on LB algorithms, nor other assumptions on the network infrastructure, nor the dynamic joining and leaving of agents do become "architecture breakers". Note that a framework itself doesn't solve the LB problem but serves as a necessary basement for LB algorithms. It abstracts the general requirements in a flexible software architecture. *Autonomy and Self-* Properties:* Increased complexity of software systems, diversity of requirements, and dynamically changing configurations, force to find new solutions based on self-organization, autonomic computing and autonomous (mobile) agents. Intelligent algorithms require autonomous agents which are advantageous in situations that are characterized by high dynamics, not-foreseeable events, and heterogeneity. *Arbitrary Configurations:* LB can be required to manage the load among local core processors on one node, as well as in a network (intranet, internet, cloud). A general LB framework must be able to cope with all these demands at the same time and offer means to abstract hardware and network heterogeneities.

In this paper, we present an extensible coordination pattern called SILBA (Self-Initiative Load Balancing Agents) with pluggable LB algorithms and autonomic (multi) agents. The main contribution of this paper is the design of a reusable and agile architectural pattern and its independence on the problem at hand. Section 2 gives a classification of existing LB algorithms. Section 2 describes the SILBA pattern and how it can be extended towards arbitrary network configurations. Section 3 presents the implementation of a LB framework based on the SILBA pattern, using a space-based middleware. In section 5, we choose some well-known algorithms (both unintelligent and intelligent) as examples, map them to SILBA, and show some benchmarks. In section 6 we summarize the results and further research work.

2 Classification of LB Algorithms

There are many different approaches that cope with the LB problem. As a first classification, we shortly listen the most important ones and classify them according to the underlying LB algorithm:

The *first* group consists of different conventional approaches without using any kind of intelligence, e.g.: Sender Initiated Negotiation and Receiver Initiated