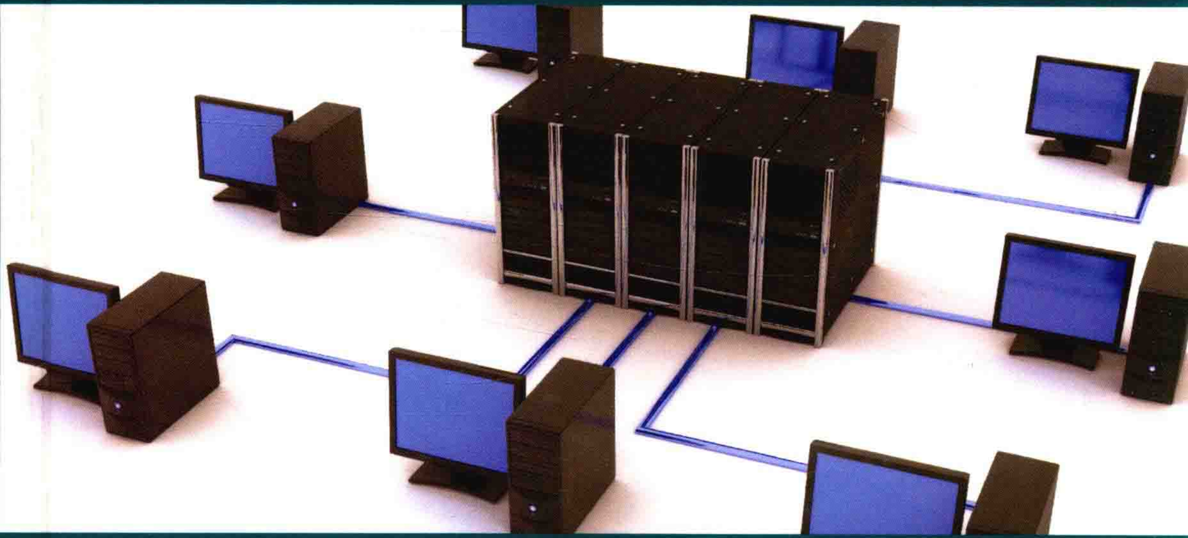
FOCUS
COMPUTER ENGINEERING SERIES



Scheduling of Large-scale Virtualized Infrastructures

Toward Cooperative Management

Flavien Quesnel

ISTE

WILEY

FOCUS SERIES

Series Editor Narendra Jussien

Scheduling of Large-scale Virtualized Infrastructures

Toward Cooperative Management

Flavien Quesnel

iSTE

WILEY

First published 2014 in Great Britain and the United States by ISTE Ltd and John Wiley & Sons, Inc.

ISTE Ltd
27-37 St George's Road
London SW19 4EU
UK

www.iste.co.uk

John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
USA

www.wiley.com

Library of Congress Control Number: 2014941926

British Library Cataloguing-in-Publication Data
A CIP record for this book is available from the British Library
ISSN 2051-2481 (Print)
ISSN 2051-249X (Online)
ISBN 978-1-84821-620-4

MIX
Paper from responsible sources
FSC® C013604
FSC www.fsc.org

Printed and bound in Great Britain by CPI Group (UK) Ltd., Croydon, Surrey CR0 4YY

Scheduling of Large-scale Virtualized Infrastructures

List of Abbreviations

ACO	Ant Colony Optimization
API	Application Programming Interface
BOINC	Berkeley Open Infrastructure for Network Computing
BVT	Borrowed Virtual Time scheduler
CFS	Completely Fair Scheduler
CS	Credit Scheduler
DOS	Distributed Operating System
DVMS	Distributed Virtual Machine Scheduler
EC2	Elastic Compute Cloud
EGEE	Enabling Grids for E-sciencE
EGI	European Grid Infrastructure
I/O	Input/Output
GPOS	General Purpose Operating System
IaaS	Infrastructure as a Service

IP	Internet Protocol
JRE	Java Runtime Environment
JVM	Java Virtual Machine
KSM	Kernel Shared Memory
KVM	Kernel-based Virtual Machine
LHC	Large Hadron Collider
MHz	Megahertz
MPI	Message Passing Interface
NFS	Network File System
NTP	Network Time Protocol
OSG	Open Science Grid
PaaS	Platform as a Service
SaaS	Software as a Service
SCVMM	System Center Virtual Machine Manager
URL	Uniform Resource Locator
VIM	Virtual Infrastructure Manager
VLAN	Virtual Local Area Network
VM	Virtual Machine
WLCG	Worldwide LHC Computing Grid
XSEDE	Extreme Science and Engineering Discovery Environment

Introduction

Context

Nowadays, increasing needs in computing power are satisfied by federating more and more computers (or nodes) to build distributed infrastructures.

Historically, these infrastructures have been managed by means of user-space frameworks [FOS 06, LAU 06] or distributed operating systems [MUL 90, PIK 95, LOT 05, RIL 06, COR 08].

Over the past few years, a new kind of software manager has appeared, managers that rely on system virtualization [NUR 09, SOT 09, VMW 10, VMW 11, APA 12, CIT 12, MIC 12, OPE 12, NIM 13]. System virtualization allows dissociating the software from the underlying node by encapsulating it in a virtual machine [POP 74, SMI 05]. This technology has important advantages for distributed infrastructure providers and users. It has especially favored the emergence of cloud computing, and more specifically of infrastructure as a service. In this model, raw virtual machines are provided to users, who can customize them by installing an operating system and applications.

Problem statement and contributions

These virtual machines are created, deployed on nodes and managed during their entire lifecycle by virtual infrastructure managers (VIMs).

Most of the VIMs are highly centralized, which means that a few dedicated nodes commonly handle the management tasks. Although this approach facilitates some administration tasks and is sometimes required, for example, to have a global view of the utilization of the infrastructure, it can lead to problems. As a matter of fact, centralization limits the scalability of VIMs, in other words their ability to be reactive when they have to manage large-scale virtual infrastructures (tens of thousands of nodes) that are increasingly common nowadays [WHO 13].

In this book, we focus on ways to improve the scalability of VIMs; one of them consists of decentralizing the processing of several management tasks.

Decentralization has already been studied through research on distributed operating systems (DOSs). Therefore, we wondered whether the VIMs could benefit from the results of this research. To answer this question, we compared the management features proposed by VIMs and DOSes at the node level and at the whole infrastructure level [QUE 11]. We first developed the reflections initiated a few years ago [HAN 05, HEI 06, ROS 07], to show that virtualization technologies have benefited from the research on operating systems, and vice versa. We then extended our study to a distributed context.

Comparing VIMs and DOSes enabled us to identify some possible contributions, especially to decentralize the dynamic scheduling of virtual machines. Dynamic scheduling of virtual machines aims to move virtual machines from one node to another when it is necessary, for example (1) to enable a system administrator to perform a maintenance operation or (2) to optimize the utilization of the infrastructure by taking into account the evolution of virtual machines' resource needs. Dynamic scheduling is still uncommonly used by VIMs deployed in production, even though several approaches have been proposed in the scientific literature. However, given the fact that they rely on a centralized model, these approaches face scalability issues and are not able to react quickly when some nodes are overloaded. This can lead to the violation of service level agreements proposed to users, since virtual machines' resource needs are not satisfied for some time.

To mitigate this problem, several proposals have been made to decentralize the dynamic scheduling of virtual machines [BAR 10, YAZ 10, MAR 11, MAS 11, ROU 11, FEL 12b, FEL 12c]. Yet, almost all of the implemented prototypes use some partially centralized mechanisms, and satisfy the needs of reactivity and scalability only to a limited extent.

The contribution of this book lies precisely in this area of research; more specifically, we propose distributed virtual machine scheduler (DVMS), a more decentralized application to dynamically schedule virtual machines hosted on a distributed infrastructure. DVMS is deployed as a network of agents organized following a ring topology, and that also cooperate with one another to process the events (linked to overloaded/underloaded node problems) that occur on the infrastructure as quickly as possible; DVMS can process several events simultaneously and independently by dynamically partitioning the infrastructure, each partition having a size that is appropriate to the complexity of the event to be processed. We optimized the traversal of the ring by defining shortcuts, to enable a message to leave a partition as quickly as possible, instead of crossing each node of this partition. Moreover, we guaranteed that an event would be solved if a solution existed. For this purpose, we let pairs of partitions merge when there is no free node left to be absorbed by a partition that needs to grow to solve its event; it is necessary to make partitions reach a consensus before merging, to avoid deadlocks.

We implemented these concepts through a prototype, which we validated (1) by means of simulations (first with a test framework specifically designed to meet our needs, second with the SimGrid toolkit [CAS 08]) and (2) with the help of real world experiments on the Grid'5000 test bed [GRI 13] (using Flauncher [BAL 12] to configure the nodes and the virtual machines). We observed that

DVMS was particularly reactive to manage virtual infrastructures involving several tens of thousands of virtual machines distributed across thousands of nodes; as a matter of fact, DVMS needed approximately 1 s to find a solution to the problem linked with an overloaded node, where other prototypes could require several minutes.

Once the prototype had been validated [QUE 12, QUE 13], we focused on the future work on DVMS, and especially on:

– Defining new events corresponding to virtual machine submissions or maintenance operations on a node;

– Adding fault-tolerance mechanisms, so that scheduling can go on even if a node crashes;

– Taking account of the network topology to build partitions, to let nodes communicate efficiently even if they are linked with one another by a wide area network. The final goal will be to implement a full decentralized VIM. This goal should be reached by the discovery [LEB 12] initiative, which will leverage this work.

Structure of this book

The remainder of this book is structured as follows.

Part 1: management of distributed infrastructures

The first part deals with distributed infrastructures.

In Chapter 1, we present the main types of distributed infrastructures that exist nowadays, and the software frameworks that are traditionally used to manage them.

In Chapter 2, we introduce virtualization and explain its advantages to manage and use distributed infrastructures.

In Chapter 3, we focus on the features and limitations of the main virtual infrastructure managers.

Part 2: toward a cooperative and decentralized framework to manage virtual infrastructures

The second part is a study of the components that are necessary to build a cooperative and decentralized framework to manage virtual infrastructures.

In Chapter 4, we investigate the similarities between virtual infrastructure managers and the frameworks that are traditionally used to manage distributed infrastructures; moreover, we identify some possible contributions, mainly on virtual machine scheduling.

In Chapter 5, we focus on the latest contributions on decentralized dynamic scheduling of virtual machines.

Part 3: DVMS, a cooperative and decentralized framework to dynamically schedule virtual machines

The third part deals with DVMS, a cooperative and decentralized framework to dynamically schedule virtual machines.

In Chapter 6, we present the theory behind DVMS and the implementation of the prototype.

In Chapter 7, we detail the experimental protocol and the tools used to evaluate and validate DVMS.

In Chapter 8, we analyze the experimental results.

In Chapter 9, we describe future work.

Contents