Journal of Network and Computer Applications (xxxx) xxxx–xxxx

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Resilient application placement for geo-distributed cloud networks

Bart Spinnewyn[a,*], Ruben Mennes[a], Juan Felipe Botero[b], Steven Latré[a]

[a] Department of Mathematics and Computer Science, University of Antwerp - iMinds, Antwerp, Belgium
[b] Department of Electronics and Telecommunications, University of Antioquia, Medellin, Colombia

ARTICLE INFO

2010 MSC:
00-01
99-00

Keywords:
Cloud computing
Quality of service
Application placement
Reliability

ABSTRACT

The strong uptake of cloud computing has led to an important increase of mission-critical applications being placed on cloud environments. Those applications often require high levels of availability coupled with guarantees on a minimum level of throughput and a maximum level of response time. To achieve the lowest response time possible, clouds are more and more decentralized, leading to a heterogeneous network of micro clouds positioned on the edge of the network and possibly interconnected by best-effort links. This heterogeneous environment introduces important challenges for the management of these clouds as the heterogeneity results in an increased failure probability. In this paper, we address these challenges by providing a resilient placement of mission-critical applications on geo-distributed clouds. We present an exact solution to the problem, which is complemented by two heuristics: a near-optimal distributed genetic meta-heuristic and a scalable centralized heuristic based on subgraph isomorphism detection. A detailed performance evaluation shows that, with the newly proposed heuristic based on subgraph isomorphism detection, we can double the amount of applications satisfying availability requirements, in cloud environments comprising over 100 nodes, while keeping the time required to calculate the solution under 20 s.

1. Introduction

Cloud computing offers computing resources as a utility: one no longer has to manage and maintain its own private computing infrastructure, instead computing infrastructure is time-shared and can be accessed on-demand. Cloud technology allows elastic scaling of resources when the demand for an application changes. While traditionally a cloud infrastructure is located within a data-center, recently, there is a need for geographical distribution. Consider for instance the case of cloud robotics (Marzio Puleri Roberto Sabella, 2016). Modern robots are capable of adapting to changing conditions, however they need a massive amount of intelligence to function properly, resulting in complex machines and control systems. Because of latency constraints, great care must be taken with the placement and management of the intelligence. On the one hand, time-critical control services such as navigation and sensor information processing must be placed close to the base station serving the working area. On the other hand, non time-critical services, such as the facility management function controlling the plant which houses the robots, can be placed remotely, as their actions do not affect real-time behavior.

Lately, this need for geo-distribution has led to a new evolution of decentralization. Most notably, the extension of cloud computing towards the edge of the enterprise network, is generally referred to as fog or edge computing (Bonomi et al., 2012). In fog computing, computation is performed at the edge of the network at the gateway devices, reducing bandwidth requirements, latency, and the need for communicating data to the servers. Second, mist computing pushes processing even further to the network edge, involving the sensor and actuator devices (Bonomi et al., 2002). Closely related to mist computing is the cloud robotics architecture put forward by Hu et al. (2012). The architecture leverages the combination of a virtual ad-hoc cloud formed by machine-to-machine (M2M) communications among participating robots, and an infrastructure cloud enabled by machine-to-cloud(M2C) communications.

Compared to a traditional cloud computing environment, a geo-distributed cloud environment is less well-controlled and behaves in an ad-hoc manner. Devices may leave and join the network, or may become unavailable due to unpredictable failures or obstructions in the environment.

Additionally, while in a data-center heterogeneity is limited to multiple generations of servers being used, there is a large spread on capabilities within a geo-distributed cloud environment. Memory and processing means range from high (e.g. servers), over medium (e.g. cloudlets, gateways) to very low (e.g. mobile devices, sensor nodes).

* Corresponding author.
E-mail addresses: bart.spinnewyn@uantwerpen.be (B. Spinnewyn), ruben.mennes@uantwerpen.be (R. Mennes), juanf.botero@udea.edu.co (J.F. Botero), steven.latre@uantwerpen.be (S. Latré).

http://dx.doi.org/10.1016/j.jnca.2016.12.015
Received 30 September 2016; Received in revised form 20 November 2016; Accepted 2 December 2016

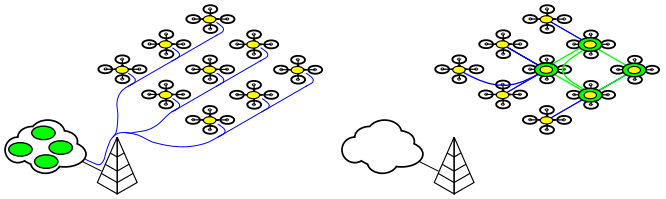
While some communication links guarantee a certain bandwidth (e.g. dedicated wired links), others provide a bandwidth with a certain probability (e.g. a shared wired link), and others do not provide any guarantees at all (wireless links).

Reliability is an important non-functional requirement, as it outlines *how* the software systems realizes its functionality (Systems and software, 2010). The unreliability of substrate resources in a heterogeneous cloud environment, severely affects the reliability of the applications relying on those resources. Therefore, it is very challenging to host reliable applications on top of unreliable infrastructure (Spinnewyn and Latré, 2015).

Moreover, traditional cloud management algorithms cannot be applied here, as they generally consider powerful, always on servers, interconnected over wired links. Many algorithms do not even take into account bandwidth limitations. While such an omission can be justified by an appropriately overprovisioned network bandwidth within a datacenter, it is not warranted in the above described geo-distributed cloud networks.

As a motivating example, consider the problem introduced by Saska et al. (2014). In the investigated scenario, the formation of multiple Micro Air Vehicle(MAVs) has to reach a desired target region in a complex environment with obstacles, while keeping predefined relative positions between the robots. The authors present a Model Predictive Control(MPC) based algorithm for maintenance of leader-follower formations of (MAVs). (MPC) is normally implemented in a centralized fashion. In large-scale interconnected systems a centralized control scheme may not be possible, and decentralized or distributed control is required (Camponogara et al., 2002). In Fig. 1, sense and actuation services (yellow) are placed on each MAV. Four distributed MPC services (green) are needed for real-time control. A first way to distribute the MPC services is to offload them to a remote infrastructure cloud (Fig. 1a). In this configuration, all communications between the actuation and sensing services happen via (M2C) communications, routed over one single base station, effectively forming a single point of failure. A second way of distribution is to execute each MPC service on a MAV (Fig. 1b). In this configuration we only make use of M2M communications. While the first configuration can benefit from the virtually infinite computing capability that resides within the remote cloud infrastructure, the second has to make do with the limited computing capabilities within the MAVs. However, while the first configuration requires all sensing data to be uploaded to the cloud in real-time, the second configuration requires no up-link capabilities at all, as the sensing data is kept within the M2M network. Additionally, while the first configuration requires the MAVs, the remote cloud infrastructure and their (M2C) interconnections to be on-line, the second configuration requires the MAVs, and their M2M interconnections to be on-line. Whichever configuration results in the best availability will ultimately depend on the failure behavior of the physical resources used. To decide which is the best configuration, we need intelligent cloud management algorithms that, next to computing resources and bandwidth limitations, consider the availability of mission-critical applications.

Current management approaches fail to provide availability guarantees in these geo-distributed cloud networks for one of following reasons. First, most approaches do not consider availability at all. Second, there are cloud management solutions that consider failures, but lack a model to calculate availability. Finally, there are approaches that model availability, but make too limiting assumptions.



(a) Offloading to a remote cloud. (b) Execution on the drones.

Fig. 1. Distribution of MPC services in a leader follower scenario for MAVs, (a) Offloading to a remote cloud., (b) Execution on the drones.

In this paper, we consider the problem of processing an initial collection of application requests upon startup of a cloud environment, which is referred to as the off-line Application Placement Problem (APP) (Jennings and Stadler, 2014). The APP has two integral parts: first, there is admission control, which selects the application requests that are accepted. Second, there is placement control, which determines how to distribute the application components in the cloud. The off-line APP differs from the on-line version, in that the application requests are all considered simultaneously, and not sequentially. We model the application requests as Virtual Networks (VNs), consisting of services and their required communication channels. The cloud infrastructure is modeled as a Substrate Network (SN) consisting of physical nodes and their interconnecting links. The problem of mapping the VNs to a SN, whilst considering failures in the SN is known as the Survivable Virtual Network Embedding (SVNE) problem (Rahman and Boutaba, 2013). Given the failure behavior of the cloud infrastructure, we solve the problem of initial distribution of application components over the cloud environment, whilst satisfying a minimum level of total availability for each application.

To the best of our knowledge, this paper is the first that provides a computationally feasible approach to place applications in a realistic and large-scale failure prone cloud environment that provides guarantees on the availability. More specifically, the contributions of this paper are four-fold. First, we present a novel approach of placing applications in a failure prone cloud environment: by adding additional replicas we are able to provide availability guarantees. This is formulated as an Integer Linear Program (ILP), which can be used to find an exact solution. Second, we propose a distributed fault-tolerant meta-heuristic based on genetic programming: a distributed set of workers concurrently search for the best placement of applications (including the definition of replicas). Third, we present a scalable centralized algorithm using the paradigm of subgraph isomorphism: this heuristic approach provides ultra-fast placement of applications with a cost in optimality. Fourth, based on an extensive performance evaluation that investigates the performance of different application types, we provide clear guidelines on when and how to apply which application placement algorithm.

The remainder of the paper is organized as follows. Section 2 discusses related works on cloud management algorithms and survivability. In Section 3 our model for availability is introduced, and in Section 4 the APP is formulated as an Integer Linear Program (ILP), which will be used to find exact solutions. In Section 5 a near-optimal distributed genetic meta-heuristic, and a scalable centralized heuristic based on subgraph isomorphism detection are presented. Section 6 presents simulations results that evaluate the proposed heuristics. Finally, Section 8 concludes the paper and summarizes the key findings.

2. Related work

In this section, the state of the art with regard to the APP in cloud environments is discussed. Early work on application placement merely considers nodal resources, such as Central Processing Unit (CPU) and memory capabilities. Deciding whether requests are accepted and where those virtual resources are placed then reduces to a Multiple Knapsack Problem (MKP) (Camati et al., 2014). An MKP is known to be NP-hard and therefore optimal algorithms are hampered by scalability issues. A large body of work has been devoted to finding heuristic solutions. For instance, Xu et al. (2010) focus on the multi-objective Virtual Machines (VMs) placement problem. They propose a genetic algorithm with fuzzy multi-objective evaluation for efficiently searching the large solution space and conveniently combin-