Scientific Report: Short Term Scientific Mission

COST Action CA15127

Sasko Ristov

April 23, 2019

1 STSM Details

STSM Title:
Resilient Services for Volatile Cloud Resources

STSM Applicant:
Dr. Sashko Ristov,
Postdoctoral University Assistant
Distributed and Parallel Systems Group, Department of Computer Science,
University of Innsbruck,
Innsbruck, Austria

Host:
Dr. Marjan Gushev,
full professor
Faculty of Computer Science and Engineering, "Ss. Cyril and Methodius University in Skopje",
Skopje, Republic of North Macedonia

Start Date:
12.04.2019

End Date:
22.04.2019

COST Action CA15127 - Resilient communication services protecting end-user applications from disaster-based failures (RECODIS)

Working group:
WG3

2 Purpose of the STSM

The purpose of this STSM was to continue with the collaboration between University of Innsbruck and Ss. Cyril and Methodius University in Skopje and jointly work on the preparation of Chapter 2.8 of the RECODIS book entitled "Resilient techniques against volatile cloud resource disruptions" within working group WG3, as well as potentially on a conference paper afterwards.

Our plan was to analyze the performance behavior and characteristics of volatile cloud resources from various cloud service providers. Based on this analysis, to overview and possibly extend the existing resilient techniques that can mitigate the probability of disruptions and their impact to various types of running applications. These techniques should be later used in scheduling algorithms in order to minimize the makespan and the economic costs of complex long-running applications when being executed on volatile cloud resources.

3 Description of the work carried out during the STSM

3.1 Problem to solve

During this STSM, we addressed the following problem to be solved. The regular On-Demand cloud resources are highly available, as claimed by the cloud service providers with an availability of at least 99.9%. The volatile cloud resources are much cheaper than the regular On-Demand, but might be reclaimed by the cloud service provider at any time, which means they have significantly lower availability and are much less reliable. This means that several disruptions can happen: (i) when a user requires volatile cloud resources, the cloud service provider might not fulfill the user request, (ii) the cloud service provider fulfills the user request, but after considerable waiting time or (iii) the volatile resources will be interrupted during runtime.

We try to minimize these disruptions and their impact when complex applications run on volatile cloud resources. Resilient techniques will reduce the overall execution time, but also the economic costs. New scheduling techniques and resilient techniques are required that will optimize execution of complex applications over volatile cloud resources.

3.2 Work carried out

Each working day of this STSM was organized in two parts: discussion and research. At the end of each working day, we discussed the discovered resilient techniques, how to systematize them and possibly extend them to be applicable for volatile cloud resources. We were also planning the work for the next day.

We divided the work into three phases: (i) determine the full life-cycle of the volatile cloud resources, with all possible events and actions, both by the user and cloud service provider, (ii) review various approaches to mitigate the risk of disruptions and their impact and (iii) apply existing resilient techniques and extend them to be applicable for complex applications over volatile cloud resources.

The summary of our results and contributions for all three activities are described in the following section.

4 Description of the main research results obtained

4.1 Volatile cloud resources life-cycle

We have detected several possible final states of a request for volatile cloud resources:

- *unfulfilled Spot request* - a Spot request is not fulfilled. The reason could be the low bid price specified by the user, which is lower than the current Spot price, or the cloud service provider does not have available volatile cloud resources when a user submits a request until the latest possible start time of the user application.

- *fulfilled Spot request after considerable waiting time (more than 1 minute)* - a Spot request is unfulfilled, but after considerable waiting time it is fulfilled. If the cloud service provider does not have available volatile cloud resources when a user submits a request, but after some time when some resources will be released, the request will be fulfilled.

- *immediately fulfilled Spot request (within four seconds)* - a Spot request is fulfilled, but after considerable waiting time it is fulfilled. If the cloud service provider does not have available volatile cloud resources when a user submits a request, but after some time when some resources will be released, the request will be fulfilled.

In case the cloud service provider fulfills the request for volatile cloud resources, the resources are provisioned and can finish in one of two final states:

- *Volatile cloud resources are interrupted* before the user terminates them, or

- *Volatile cloud resources are not interrupted* and the user will terminate them.

We have to note that the requests for regular On-Demand cloud resources are always fulfilled immediately and those resources are never interrupted by the cloud service provider, but the user has to terminate them. This means that whenever the request for volatile cloud resources finishes in one of the first two final states or the first final state if the request is fulfilled, we have to deal with a disruption of volatile cloud resources.

4.2 Modelig the volatile cloud resources

All these events lead to three important parameters that should be known by the user in order to develop resilient techniques:

- *fulfillment rate* - the percentage between 0 and 100 that presents the portion of all requests for volatile cloud resources that will be deployed,
- *waiting time* - period of time that the user waits for the cloud service provider to fulfill the request
- *frequency of interruption* - the percentage between 0 and 100 that presents the portion of all interrupted volatile cloud resources. Just to note that this percentage is calculated only over the fulfilled requests. For example, from 10 submitted requests, the fulfillment rate of 80% leads to 8 fulfilled requests, and frequency of interruption of 25% means that 2 volatile cloud resources will be interrupted.

Just to note that this definition is generalized and the On-Demand instances can be modeled with these parameters, as well, i.e. their fulfillment rate is always 100%, while waiting time is 0 and frequency of interruption is 0%.

4.3 Available data for volatile cloud resources is not enough

Cloud service providers help the users to better understand the behavior of volatile cloud resources and publish a lot of information about their volatile resources. For example, a full pricing history for all Amazon's volatile cloud resources in all regions for the last three months is publicly available.

Still, other important information is only partially available or even completely unavailable. For example, the average frequency of interruption for the last calendar month is published, clustered in five groups (bellow 5%, $5-10\%$, $10-15\%$, $15-20\%$, and more than 20%). Neither the reason for interruption is known((*no-capacity* or *low-bid-price*)), nor the availability zone within the region for which the frequency of interruption holds. The information about the fulfillment rate of requests for volatile cloud resources, as well as the waiting time for fulfillment is not publicly available. Users cannot estimate the values of the three parameters (fulfillment rate, waiting time and frequency of interruption) as no correlation exists between them [1].

4.4 Resilient techniques for volatile cloud resources

we have determined several resilient techniques for various types of applications, which will be elaborated in Chapter 2.8 of the RECODIS book:

- Reactive resilient techniques for time-insensitive applications (e.g. switch to On-Demand after interruption or waiting time longer than 4 seconds). There are two approaches:
  - minimize the economic costs (cheaper) or
  - minimize the delay (faster).
- Proactive bidding strategies to alleviate volatile cloud resources disruptions due to low-bid-price (e.g. increase of spot price).
- Resilient techniques for complex applications (with dependencies between tasks) to overcome volatile cloud resources disruptions:
  - task retry,
  - alternate resource,
  - checkpoint-restore, and
  - task replication.

5 Future collaboration with the Host institution

The dynamic scheduling algorithms are complex and both institutions will work closely on its implementation as a follow up in the VolatileSim simulator developed by the University of Innsbruck. The discussions will be organized via conference calls.

After finishing the implementation of the scheduling algorithm and its evaluation, the collaboration will be extended in several directions:

- Resilient techniques for complex applications with streaming data between tasks, which will allow to simulate the IoT, Edge, Fog and Dew applications
- Resilient techniques for containers, micro-services and serverless applications
- Describe the resilient techniques together with the workflow description (e.g. extend AGWL[1] for resilience)

6 Foreseen publications/articles resulting from this STSM

The main goal of this STSM was to work jointly for Chapter 2.8 of the RECODIS book. However, after publishing the VolatileSim simulator developed by the University of Innsbruck, the plan is to model and implement the resilient techniques in VolatileSim, which should result in a conference or a journal paper.

7 Acknowledgement

We would like to thank the COST action for granting this STSM and performing this research, which will help the STSM applicant in writing Chapter 2.8 of the RECODIS book, as well as extend the networking with other members of the COST Action.

References

[1] Thanh-Phuong Pham and Sasko Ristov and Thomas Fahringer, "Performance and behavior characterization of Amazon EC2 spot instances," in *IEEE Cloud 2018*, 2018, pp. 73–81.

[1] http://dps.uibk.ac.at/projects/agwl/