Interaction-awareness for self-adaptive volunteer computing

Abdessalam Elhabbash, Rami Bahsoon, Peter Tino
School of Computer Science
University of Birmingham
United Kingdom
{a.elhabbash, r.bahsoon, p.tino}@cs.bham.ac.uk

***Abstract*— In this paper, we contribute to a self-adaptive approach, namely interaction-awareness which adopts self-aware principles to dynamically manage and maintain the knowledge on the interactions between volunteer services in the volunteer computing paradigm. Such knowledge can inform the adaptation decisions; leading to increase in the precision of selecting and composing services. We evaluate the approaches using a volunteer storage composition scenario. The evaluation results show the advantages of dynamic knowledge management in self-adaptive VC in selecting dependable services and satisfying higher number of requests.**

Keywords—dependability; service composition; self-adaptive; self-aware

I. INTRODUCTION

Volunteer Computing (VC) is an emerging distributed computing paradigm in which users make their own resources available to others enabling them to do distributed computations and/or storage. In the literature, many VC approaches have been proposed, e.g. [1], to enable cost-effective large scale computation and sharing for storage, leveraging on spare resources that can be available and idle on the users' edge computing devices. Volunteered resources can be composed together to satisfy users' requests in many service-oriented applications, a practice which we term as volunteer service composition (VSC). Engineering Volunteer Services (VSs) calls for novel self-adaptive approach for adaptively managing the processes of selecting, composing, and allocating VSs and underlying resources. The approach shall address the following fundamental requirements [2]:

- Resources-awareness: the contributed resources should be composed and/or allocated to users, achieving both maximum utilization and minimum waste with minimum computation time.
- Availability-awareness: Resources availability tends to be uncertain in VC because the publishers contribute their resources during the time intervals in which they do not need those resources.
- Dilution of control: VSs can be offered and withdrawn at any time, which makes the Service Level Agreements less stringent as when compared to commercial services.
- Dependability-awareness: Because of the dilution of control requirement, dependability information of the services, in terms of the level of providing the promised resources, should be collected and used in VS composition/allocation approaches.

In [3] the authors have reported that the independence assumption on hosts' performance is not always valid. That is, hosts performance can be correlated. Based on that, the awareness of such correlation enables reasoning on selecting hosts that exhibit satisfying performance in case when the hosted services need to interact with each other when composed to satisfy a certain request.

Recently, self-awareness concept has been receiving more attention in computing systems [4]. Self-awareness can provide self-adaptive systems with primitives for proactive management and behavioral control at runtime. It can also improve both the accuracy and quality of adaptation. This may in turn converge the system towards more stable states.

In [5], we proposed a self-aware framework to enable self-adaptation in VC. As part of the framework, a self-adaptive approach, namely stimulus-awareness, has been proposed to enable basic self-adaptation capabilities. In this paper we take this work further. We propose the interaction-awareness approach which considers the knowledge on the previous interactions between the services. The approach treats knowledge as "moving target" that can change and evolve over time and use this information to better inform the adaptation. For this purpose, we make novel use of dynamic histograms, which are constructs that dynamically approximate data distributions at runtime [6], to capture the evolving knowledge on the services' interactions. This can consequently improve the quality and precision of adaptation in dynamic and uncertain environment.

II. INTERACTION-AWARE VSC

A. Basic Definitions

Def 1. (Volunteer Service). A volunteer service VS_i, is a 3-tuple (stg_i, T_i, sec_i) where stg_i is the volunteered storage space, T_i is the time interval $[a_i, b_i]$ in which the VS_i is available, and sec_i is the security level promised by the service.

Def 2. (Subscriber's Request). A subscriber's request R is a 3-tuple (stg^R, T^R, sec^R), where stg^R denotes the required storage, $T^R = [a^R, b^R]$ is the required time interval in which stg^R is required, and sec^R is the required security level where $0 \leq sec^R \leq sec_{max}$ and $sec^R, sec_{max} \in \mathbb{N}$.

Def 3. (Composite Service). Given a subscriber's request R, a Composite Service CS is a set of VSs, $\{VS_1, VS_2, \ldots, VS_k\}$ that satisfy the subscriber's requirements.

Def 4. (Storage utility U^P_{stg}). A measure of the promised storage space contributed by a service to satisfy a certain request.

Def 5. (Availability-Time utility U^P_{time}). A measure of the time period in which a service promises to be available.

Def 6. (Security utility U^P_{sec}). A measure of the promised security level contributed to satisfy a certain request.

Def 7. (VS dependabilities). A service VS_i is considered to be *dependable* if VS_i provides the storage, availability, and security it promises. Mathematically, the storage dependability is expressed as in (1).

$$D_{stg}(VS_i) = \begin{cases} \dfrac{U^P_{stg}(VS_i) - U^A_{stg}(VS_i)}{U^P_{stg}(VS_i)}, & U^A_{stg}(VS_i) < U^P_{stg}(VS_i) \\ 1, & Otherwise \end{cases} \quad (1)$$

where U^A_{stg} is the actual utility. The availability time dependability, D_{time}, and the security dependability, $D_{sec}(VS_i)$, are expressed similarly.

B. Composition approach

The aim of the interaction-aware approach is to consider the past interactions between the services in order to capture the correlation that exist between services. In other words, the interaction-aware approach aims to predict which services are most appropriate to be composed together to satisfy a request. To achieve that, we maintain a matrix of dynamic histograms for the services which has the form:

$$DHM_{stg} = \begin{array}{c} \\ VS_1 \\ VS_2 \\ \vdots \\ VS_n \end{array} \begin{array}{c} \begin{array}{cccc} VS_1 & VS_2 & \ldots & VS_n \end{array} \\ \begin{pmatrix} - & dh_{12} & \ldots & dh_{1n} \\ dh_{21} & - & \ldots & dh_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ dh_{n1} & dh_{n2} & \ldots & - \end{pmatrix} \end{array}$$

where VS_1, VS_2, … , VS_n are the available services, dh_{ij} is the dynamic histogram that maintains the dependabilities of VS_i when composed with VS_j. Similarly, two matrices are maintained for the time availability (DHM_{time}) and security (DHM_{sec}) attributes. Initially each dynamic histogram contains one bucket, then the dynamic histogram evolves by dividing/merging buckets as the dependabilities' data points arrive, where buckets represents the time intervals during which the service has been used. Now, when a request is submitted, the dynamic histograms will be used to estimate the interaction dependabilities between services. To do so, the dependability in each bucket is estimated by finding buckets that intersect with the request time interval and averaging the values of the data points in those buckets. After that, the system applies a greedy approach to select one service per iteration using (2) until the request is satisfied. The output of the greedy selection is a CS that satisfies the requests or an empty CS is case when the request cannot be satisfied.

$$\begin{aligned} & maximize\{U_{Stg}(VS_i),\ U_{time}(VS_i),\ U_{sec}(VS_i) \\ & \qquad D_{stg},\ D_{time},\ D_{sec}\} \\ & s.t. \quad U_{sec}(VS_i) > 0 \end{aligned} \quad (2)$$

C. Interaction-aware adaptation

The self-adaptability in the interaction-aware approach is two-fold (1) Reactive adaptation and (2) Proactive adaptation. The former is applied when a change in the promised utilities is reported while the latter is applied when a violation of the promised utilities is predicted, using the dependabilities information stored in the dynamic histograms.

III. Experimental Evaluation

We conduct experiments in order to evaluate the performance of the interaction-aware approach and compare it with the baseline approach, the stimulus-awareness. Fig. 1(a) shows that the average throughput is almost the same in the initial period of time. After a while of accumulating the knowledge, the average throughput in the interaction-aware case gets higher than the stimulus-aware. Fig. 1(b) shows that the stimulus-aware approach has the least time cost and that the interaction-aware approach increases linearly over time.

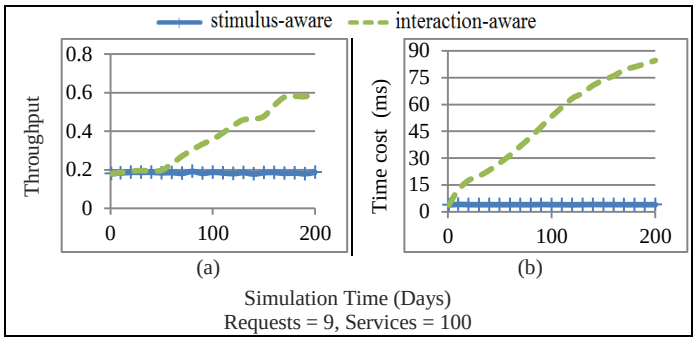

Fig. 1. Comparison in throughput and time cost.

IV. Conclusion

We have contributed to a self-adaptive approach, which make novel use of the principles of self-awareness and dynamic histograms to dynamically manage knowledge in self-adaptive VC. The approach is able to capture the evolving knowledge on the services interactions to reason about the reactive and proactive adaptation decisions. The experimental results show that the interaction-aware approach can bring to a self-adaptive application the advantages of satisfying more requests but accompanied with overhead.

References

[1] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, "SETI@ home: an experiment in public-resource computing," *Communications of the ACM*, vol. 45, pp. 56-61, 2002.

[2] S. Sebastio, M. Amoretti, and A. L. Lafuente, "AVOCLOUDY: a simulator of volunteer clouds," *Software: Practice and Experience*, 2015.

[3] M. Bakkaloglu, J. J. Wylie, C. Wang, and G. R. Ganger, "On correlated failures in survivable storage systems," DTIC Document2002.

[4] S. Kounev, X. Zhu, J. O. Kephart, and M. Kwiatkowska, "Model-driven algorithms and architectures for self-aware computing systems (Dagstuhl Seminar 15041)," *Dagstuhl Reports*, vol. 5, 2015.

[5] A. Elhabbash, R. Bahsoon, P. Tino, and P. R. Lewis, "Self-Adaptive Volunteered Services Composition through Stimulus-and Time-Awareness," in *Web Services (ICWS), 2015 IEEE International Conference on*, 2015, pp. 57-64.

[6] D. Donjerkovic, Y. Ioannidis, and R. Ramakrishnan, "Dynamic Histograms: Capturing Evolving Data Sets," in *Data Engineering, 2000. Proceedings. 16th International Conference on*, 2000, pp. 86-86.