Towards a Self-healing Multi-agent Platform for Distributed Data Management

Arles Rodríguez[1,2](✉), Jonatan Gómez[2], and Ada Diaconescu[3]

[1] Fundación Universitaria Konrad Lorenz, Bogotá, Colombia
arlese.rodriguezp@konradlorenz.edu.co
[2] ALIFE Research Group, Universidad Nacional de Colombia, Bogotá, Colombia
jgomezpe@unal.edu.co
[3] Telecom ParisTech, IMT, Paris-Saclay University, Paris, France
ada.diaconescu@telecom-paristech.fr

Abstract. We demonstrate a self-healing multi-agent simulation platform for distributed data-management tasks, including data collection and synchronisation. Collective tasks can be simulated within two types of environments: uncharted terrains with various obstacles, and computing networks with different complex topologies. Agents explore their environment, collect and update local data, and exchange data with agents that they encounter, until the collective task is completed. We have previously implemented several agent exploration algorithms and evaluated their performance in terms of completion speed (essential when agents may fail) and resource overheads (essential in constrained environments). Here, we focus on the agents' ability to self-heal, via local replication, so as to ensure task completion. We focus on computing network environment, where software replication is more feasible. Envisaged applications include data management in computing clouds, distributed databases, sensor networks, robot swarms and the Internet of Things.

Keywords: Multi-agent · Simulation · Self-healing · Data-management tasks

1 Introduction

Data centres, server farms and clouds are distributed systems consisting of a myriad of computing resources interconnected via a network, and coordinating their actions, transparently to users, in order to accomplish various tasks [1]. Such systems are difficult to manage – e.g. software updates, failed component replacements – and downtimes can cost companies in the order of thousands of dollars per minute [2]. Autonomic Computing [2,3] drew inspiration from nature and proposed to enable computing systems to self-manage, minimising expensive and error-prone human intervention. Notably, self-healing allows systems to recover and pursue their tasks despite failures [4,5].

The proposed demonstration presents a multi-agent simulator for exploring decentralised self-healing functions and evaluating robustness in distributed systems. Within this simulator, we model and experiment with failure-prone agents

Y. Demazeau et al. (Eds.): PAAMS 2017, LNAI 10349, pp. 350–354, 2017.
DOI: 10.1007/978-3-319-59930-4_36

which cooperate to achieve collective data-management tasks, such as data collection from uncharted terrains [6] and data synchronisation across complex networks [7]. We evaluated different agent exploration algorithms, e.g. based on random movement, swarm intelligence and Lévy walks. In uncharted terrain environments, results show that a pheromone-based exploration approach ensures the fastest task completion and hence better robustness in case of agent failures. In complex network environments, the same pheromone-based algorithm performs best for most network topologies (e.g. Random, Community, or Small World), yet random exploration is better in topologies with large hubs – i.e. with large values for the standard deviation of the betweenness centrality of their nodes (e.g. some Scale Free or Hub & Spoke topologies).

The present work proposes a self-healing function based on local agent replication. In short, each distributed node keeps track of agents departing for neighbouring nodes. Upon arrival at a new node agents send a confirmation message back to their departing node, which consequently stops tracking them. When a node does not receive a confirmation message from a departed agent within a *time-out* interval, it creates a new agent and injects its local state (i.e. local data) into it. If a confirmation message arrives late (i.e. after the time-out and after a replica has already been created) the node removes the next agent that arrives at the node (after copying its data) and updates its local time-out (i.e. learning). Details and results are available from the accompanying paper[1].

The simulator provides results on task success rates, completion speed and replication overheads (e.g. extra memory and communication). We believe that these findings and platform can help to experiment with various multi-agent solutions for a wide variety of data-intensive distributed systems.

2 Platform Purpose and Implementation

The presented simulation platform[2] allows developing various multi-agent data-management solutions, with self-healing capabilities, and evaluating their performance and robustness in different distributed environments. The simulator is implemented in Java, based on the multi-agent platform in [8] – with agents implemented via a family of classes, and running in separate Threads. In demonstrated scenarios the agents are specified as in [7] in terms of exploration algorithms, data management and inter-agent exchanges. The environment is defined as another extensible family of classes that allows agents to interact (e.g. a bi-dimensional terrain or a complex network).

Simulation metrics are defined using the Observer design pattern, which separates simulations from generated metric reports. These reports allow obtaining various statistics (e.g. box-plots and histograms), including the number of steps required for task completion, the number of message exchanges, task success

[1] "Replication-based Self-healing of Mobile Agents Exploring Complex Networks" – submitted to PAAMS 2017.

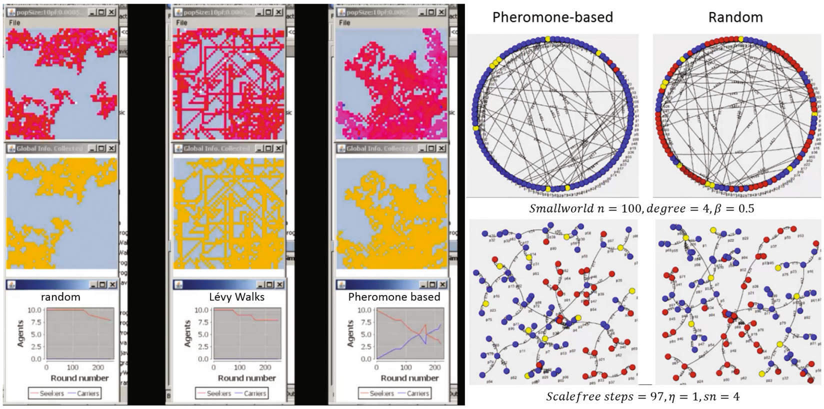
[2] http://www.alife.unal.edu.co/%7Eaerodriguezp/networksim/.

rates, or the evolution of agent numbers over time. The simulator's statistics module can also be extended and modified to develop custom metrics.

3 Demonstration

The demonstration shows different types of simulations that were developed using the proposed platform. Firstly, as in Fig. 1a, we provide a simulation of failure-prone agents with different strategies for exploring a bi-dimensional terrain [6]. In Fig. 1a, the upper part shows the agents' terrain coverage (purple traces), the middle part shows the terrain information collected (yellow marks), and the bottom part shows graphs plotting the live agents (failing with a certain probability) against the simulation round number. This simulation allowed us to determine which exploration strategies are more robust in case of agent failures, faster in terms of simulation rounds, and lighter in terms of resource overheads.

Secondly, as in Fig. 1b, we present a simulation of agents (in yellow) collecting and synchronising data within various complex networks [7]. Locations explored by agents are in blue and locations not explored in red. Implemented topologies include Small World, Scale Free and Community (using JUNG [9]), as well as simpler ones such as Hub & Spoke, Lattice, Line and Circle (for testing extreme conditions). This allows us to profile the performance and dependability of different agent exploration strategies against each network topology, for different agent failure rates. Results show a correlation between these evaluation metrics and the standard deviation of the node betweenness centrality – intuitively, pheromone-based exploration techniques are hindered by topologies featuring large hubs and few alternative routes, since hubs get pheromone-marked and become temporarily inaccessible for further passing.



a) Failing agents exploring a bi-dimensional space

b) Failing agents collecting data from some complex networks

Fig. 1. Different simulations generated

Thirdly, we extend the previous simulation by endowing agents with self-healing capabilities. In this case, results show that agents can successfully complete the collective task even in the presence of high-failure rates (which was not the case without self-healing), while inducing limited local overheads.

4 Conclusions and Future Work

This demonstration shows an agent-based simulator for modelling distributed tasks. Agents are modelled to carry internal states, to explore their environments (either continuous surfaces or complex networks), to perform local data-management tasks, and to communicate with each other when they meet.

The main contribution of this simulator is to help design and evaluate different decentralised data-management solutions, applicable to various distributed environments, with different characteristics (e.g. diverse tasks, resource constraints, performance requirements, or agent failure rates).

The simulator collects metrics that enable statistic analysis, which are critical for profiling new agent designs. So far, this allowed us to determine the best agent exploration strategy for performing a distributed task in different types of terrains and network topologies, with different agent failure rates.

Future work will model and simulate new strategies for recovering from node failures and corrupt data collection. Our objective is to provide a theoretical and experimental base for developing real applications for different distributed environments – e.g. data collection and replication in clouds, clusters and the Internet of Things. The source code and results obtained are available at http://www.alife.unal.edu.co/%7Eaerodriguezp/networksim/.

References

1. Tanenbaum, A., Steen, M.V.: Distributed Systems: Principles and Paradigms. Prentice-Hall, Upper Saddle River (2006)
2. Lalanda, P., Mccann, J.A., Diaconescu, A.: Autonomic Computing: Principles, Design and Implementation. Springer, Heidelberg (2013)
3. Kephart, J.O., Chess, D.M., Jeffrey, O., David, M.: The vision of autonomic computing. Computer **36**, 41–50 (2003)
4. Hu, J., Gao, J.I., Liao, B.S., Chen, J.J., Jun, W.: Multi-agent system based autonomic computing environment. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics, vol. 1, pp. 105–110 (2004)
5. Bisadi, M., Sharifi, M.: A biologically-inspired preventive mechanism for self-healing of distributed software components. In: The Second International Conference on Advanced Engineering Computing and Applications in Sciences, ADVCOMP 2008, pp. 152–157 (2008)
6. Rodriguez, A., Gomez, J., Diaconescu, A.: Foraging-inspired self-organisation for terrain exploration with failure-prone agents. In: 2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems, pp. 121–130. IEEE, October 2015

7. Rodriguez, A., Gomez, J., Diaconescu, A.: Exploring complex networks with failure-prone agents. In: Verlag, S. (ed.) 15th Mexican International Conference on Artificial Intelligence, MICAI 2016. LNCS (2016)
8. Gomez, J.: Unalcol agents (2016). https://github.com/jgomezpe/unalcol/tree/master/agents/src/unalcol/agents
9. White, S.: Analysis and visualization of network data using JUNG. J. Stat. Softw. **VV**, 1–35 (2005)