Simulating Large-scale Social Sensing based Edge Computing Systems with Heterogeneous Network Configurations

Nathan Vance, Ryan Mackey, Daniel (Yue) Zhang, Dong Wang
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN, USA
nvance1@nd.edu, rmackey1@nd.edu, yzhang40@nd.edu,dwang5@nd.edu

***Abstract*—Research in the field of Social Sensing Edge Computing (SSEC) faces a challenge in that most systems are developed on a small set of hardware and are never tested on large-scale networks. To address this issue, we present a Social Sensing Edge Computing Simulation Platform (SSEC_SP) that is used to evaluate how SSEC systems scale to millions of devices. We demonstrate the platform by simulating several SSEC configurations to optimize Quality of Service (QoS) for large numbers of devices, while asserting that simulated results are consistent with the hardware implementations.**

I. Introduction

Social Sensing Edge Computing (SSEC) is an application of edge computing [1] to the field of social sensing [2], [3]. Examples of social sensing applications include disaster response [4], intelligent transportation systems [5], environment monitoring [6], and urban sensing [7]. In SSEC, edge devices are incentivised to collect sensing data, process it, and send the results to an aggregating server [8]. These systems are often implemented using complex network topologies such as utilizing "fog servers" as an intermediate layer between edge devices and backend servers [9], [10], or by parallelizing calculations among several edge devices [11].

One issue with developing a complex network topology for a SSEC system is that it is not immediately obvious what the system's bottlenecks will be as it scales to millions of devices [12]. For example, consider the case of using fog servers to alleviate some of the burden from the backend server. In such a system, it can be difficult to accurately predict how many fog servers are needed to optimize for QoS, especially when the devices are heterogeneous. If too many fog servers are deployed, then the backend server may not be able to keep up with the demand necessary to coordinate them. However, if too few fog servers are deployed, then they may be overwhelmed by the edge devices. Furthermore, if additional layers of servers are added to the hierarchy, then QoS may be unacceptably degraded by the increased latency [13].

In this demo, we present the Social Sensing Edge Computing Simulation Platform (SSEC_SP). This platform enables researchers to evaluate SSEC solutions and detect bottlenecks as the systems scale to large numbers of devices. We implement SSEC_SP using ns-3 [14] as a backend and abstracting SSEC primitives.

II. System Overview

The SSEC_SP platform will model interactions between networking devices common in edge computing scenarios:

- **Hosts:** Each computer on the network involved in SSEC.
- **Cloud Server:** The host that is responsible for defining tasks, aggregating results, and issuing incentives.
- **Edge Device:** A host that collects data and engages in some processing.
- **Fog Server:** An intermediate host between edge devices and the cloud server that coordinates edge devices for processing data.
- **Networking Equipment:** The connections, switches, and routers that enable communication between hosts, each with a bandwidth and communication latency.
- **Data Communication:** A transfer of data between hosts using networking equipment. This is the base class of interactions between hosts.

The platform will be flexible, allowing the researcher to design a topography, script behavior on arbitrary classes of devices, and evaluate QoS under various conditions. It captures the most common sources of latency:

- **Networking latency:** The overhead incurred by sending data over the network. Each network connection and device has an associated bandwidth and latency, allowing us to calculate the delay caused by the network.
- **Computational latency:** For each host, the model defines the time required to process each datum and the number of processes that may occur in parallel. Using these parameters we calculate the delay caused by data processing.

Each class of devices in the SSEC model have scripted behavior describing how they react to receiving data of a specified descriptor, size, and origin. These scripts produce output data of a specified descriptor, size, and destination, and also reports the computational overhead incurred in the data processing. For example, when a fog server receives 5

MB of data with a descriptor "raw sensor data" from an edge device, its script may send 1 MB of data to 5 edge devices with the descriptor "distributed computing". The script reports to SSEC_SP that this operation required 2 units of computation overhead. SSEC_SP multiplies this overhead by that device's speed (in units of computation overhead processed per second) to obtain the computational latency. Note that SSEC_SP does not actually transfer data between simulated hosts, but rather just metadata. This is so that the simulation itself can scale.

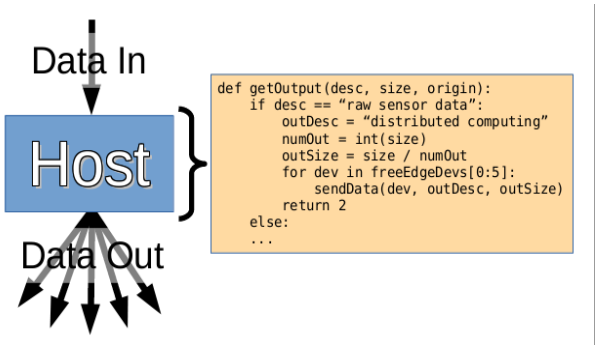

Fig. 1. Data Flow in Simulation

III. The Demo

The SSEC_SP system runs on a laptop. It reads in configuration files describing:

- **Hosts:** The various classes of hosts and their configurations. In particular, configurable parameters include networking bandwidth, networking latency, computational speed, max parallel calculations, and the extent of random variation in each of the above.
- **Scripts:** These define the behavior of classes of hosts when they receive data. Run-time parameters to a script include input data origin, input data size, input data descriptor (used to simulate decision making based on data content). Outputs from a script include: output data destination(s) (could be zero for no output), output data size, output data description, and computational overhead.
- **Topography:** The quantity of each type of host and the connectivity between them.
- **Tasks:** The tasks and frequency of their issuance. These can originate from any class of hosts, e.g. the edge devices for a bottom up system, or the cloud server for a top down system.

In the configuration files, the hosts, topography, and tasks are declared in XML, and the scripts are written in Python. This setup affords maximum flexibility for the researcher to describe the complex specifics of the modeled system.

After simulating the system, SSEC_SP will output:

- Statistics describing the latency for completing tasks
- Statistics describing the saturation of each host and network device during the simulation

IV. Conclusion

In conclusion, we present a social sensing edge computing simulation platform (SSEC_SP) that can simulate complex social sensing networks. We demonstrate that SSEC_SP can be used to evaluate the scalability of an edge computing topography with heterogeneous devices.

Acknowledgement

This research is supported in part by the National Science Foundation under Grant No. CBET-1637251, CNS-1566465 and IIS-1447795 and Army Research Office under Grant W911NF-17-1-0409. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[2] D. Wang, T. Abdelzaher, and L. Kaplan, *Social sensing: building reliable systems on unreliable data*. Morgan Kaufmann, 2015.

[3] D. Wang, B. K. Szymanski, T. Abdelzaher, H. Ji, and L. Kaplan, "The age of social sensing," *arXiv preprint arXiv:1801.09116*, 2018.

[4] Y. Zhang, N. Vance, D. Zhang, and D. Wang, "Optimizing online task allocation for multi-attribute social sensing," in *The 27th International Conference on Computer Communications and Networks (ICCCN 2018)*. IEEE, 2018.

[5] D. Wang, T. Abdelzaher, and L. Kaplan, "Surrogate mobile sensing," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 36–41, 2014.

[6] D. Y. Zhang, Y. Ma, Y. Zhang, S. Lin, X. S. Hu, and D. Wang, "A real-time and non-cooperative task allocation framework for social sensing applications in edge computing systems," to appear in Real-Time and Embedded Technology and Applications Symposium (RTAS), 2018 IEEE. IEEE, 2018, accepted.

[7] J. Zhang and D. Wang, "Duplicate report detection in urban crowdsensing applications for smart city," in *Smart City/SocialCom/SustainCom (SmartCity), 2015 IEEE International Conference on*. IEEE, 2015, pp. 101–107.

[8] D. Y. Zhang, C. Zheng, D. Wang, D. Thain, X. Mu, G. Madey, and C. Huang, "Towards scalable and dynamic social sensing using a distributed computing framework," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017, pp. 966–976.

[9] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proceedings of the 2015 Workshop on Mobile Big Data*, ser. Mobidata '15. New York, NY, USA: ACM, 2015, pp. 37–42. [Online]. Available: http://doi.acm.org/10.1145/2757384.2757397

[10] M. T. Al Amin, T. Abdelzaher, D. Wang, and B. Szymanski, "Crowd-sensing with polarized sources," in *Distributed Computing in Sensor Systems (DCOSS), 2014 IEEE International Conference on*. IEEE, 2014, pp. 67–74.

[11] D. Y. Zhang, D. Wang, Y. Zhang, N. Vance, and S. Mike, "On scalable and robust truth discovery in big data social media sensing applications," *IEEE Transaction on Big Data (in press)*, 2018.

[12] D. Wang, M. T. Amin, S. Li, T. Abdelzaher, L. Kaplan, S. Gu, C. Pan, H. Liu, C. C. Aggarwal, R. Ganti *et al.*, "Using humans as sensors: an estimation-theoretic perspective," in *Information Processing in Sensor Networks, IPSN-14 Proceedings of the 13th International Symposium on*. IEEE, 2014, pp. 35–46.

[13] Y. Zhang, N. Vance, D. Zhang, and D. Wang, "On opinion characterization in social sensing: A multi-view subspace learning approach," to appear in Distributed Computing in Sensor Systems (DCOSS), 2017 International Conference on. IEEE, 2017.

[14] NS-3, "Network simulator 3." [Online]. Available: https://www.nsnam.org