Energy-aware Placement of Device-to-Device Mediation Services in IoT Systems

Abdessalam Elhabbash[1][0000−0002−9463−0446]
and Yehia Elkhatib[2][0000−0003−4639−436X]

[1] School of Computing and Communications, Lancaster University, United Kingdom
[2] School of Computing Science, University of Glasgow, United Kingdom
a.elhabbash@lancaster.ac.uk, yehia.elkhatib@glasgow.ac.uk

Abstract. Internet-of-Things (IoT) systems are becoming increasingly complex, heterogeneous and pervasive, integrating a variety of physical devices, virtual services, and communication protocols. Such heterogeneity presents an obstacle especially for interactions between devices of different systems that encounter each other at run time. Mediation services have been proposed to facilitate such direct communication by translating between messaging protocols, interfacing different middlewares, etc. However, the decision of where to place a mediation service within an IoT topology has repercussions and is in some cases critical for satisfying system objectives. In this paper, we propose an integer linear programming solution to optimize the placement decision specifically in terms of energy consumption. Our solution takes into account the energy consumed by each interaction at each device along the data transfer paths. Through simulations that use topologies of real-world IoT systems, we show the effect of our approach on energy consumption, messaging delay, and placement decision time. Our algorithm outperforms a state-of-the-art solution in terms of reducing energy consumption by almost a third in large-scale typologies. We also demonstrate the feasibility of our approach in terms of overhead.

Keywords: Energy consumption · Internet of Things · Cyber physical systems · Mediator · Middlebox · Sustainable computing

1 Introduction

The Internet of Things (IoT) and Cyber-Physical System (CPS) paradigms connect a variety to devices in order to form a system that is capable of monitoring and controlling its environment. The benefits of this paradigm span across several areas such as smart cities [24], smart buildings [25] and environmental monitoring [26], among others. These tangible benefits have given rise to the production of vast numbers of IoT devices with an expected growth from 8.74 billion in 2020 to more than 25.4 billion by 2030 [1].

Interoperability between IoT devices is a major challenge when using device-to-device (D2D) communication. IoT industry producers tend to develop their

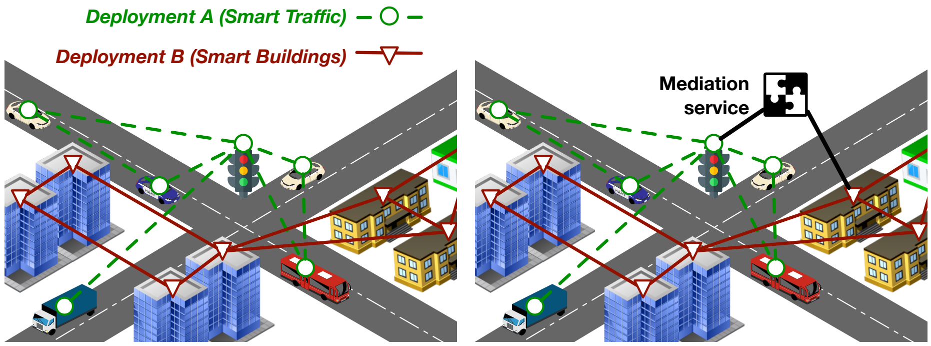

Fig. 1: An example of IoT and CPS deployments sharing the same environment but not being able to intercommunicate due to heterogeneity arising from the use of different communication protocols, service semantics, message formats, middleware software, etc. Deploying a mediation service (see figure on the right) enables interoperation between different deployments.

own APIs and protocols to enable connectivity of their devices given the constraints of their service [33]. This has created a large space of highly heterogeneous devices. However, the difference of APIs, messaging models and message formats complicates direct interaction. For example, the CoAP [32] protocol adopts a client-server messaging model and a maximum message size of 1152 bytes whereas MQTT [7] adopts a publish-subscribe model of messages upto $\approx$260 megabytes. Therefore, a device that uses CoAP protocol will not be able to interoperate with another that uses MQTT (see illustration in Fig. 1).

Consider for example the case of a fire fighting emergency team in a smart building. In this scenario, the rescue crew may need to install their equipment in the site and interact with the smart building network to collect situational awareness data. It may not be attainable or convenient to adopt a cloud-based architecture in this case due to unavailability or high delays. In this case, direct interaction is required to interconnect the rescue equipment with the building devices and, thus, a mediator is inevitably necessary.

A solution to cope with the heterogeneity issue is to employ a middlebox to bridge between devices and abstract their functional semantics. The middlebox will reside somewhere in the network as a mediation service and translates between the messaging models of different protocols. Examples include network intent mediation [15], the FIESTA-IoT directory service [31], and the (Data eXchange Mediator Synthesizer) DeXMS framework [9].

However, a notable question that the literature on mediation services does not answer is *where* to place the mediator in the network. This question has not yet been thoroughly tackled by the IoT community, though a method for optimizing the end-to-end delay between the interacting devices has recently been proposed [12]. We argue that energy consumption is a substantial factor to consider in such cases for two reasons. First, efficient energy consumption is crucial to maintain device functionality for the longest period of time possible, especially that some IoT devices have non-rechargeable power sources. Second, efficient energy consumption contributes to the principle of designing sustainable computing solutions.

In this paper, we develop a method that utilizes the network structure to compute the placement of mediation services in order to minimize the energy consumed by the interactions between IoT devices. Our method formulates the placement problem as an integer linear programming (ILP) problem and produces the optimal placement given the interaction load and bandwidth constraints. In this sense, the proposed method is adaptive as it allows placement recalculation whenever the data size and/or available bandwidth change. We compare our proposed method to the delay-optimizing method in recent literature [12] and with a naïve baseline method of random placement. The results show that our adaptive method achieves minimal energy consumption for different IoT network topologies.

Overall, this paper makes the following contributions:

- We formulate the placement of a mediation service as an ILP problem (§3.1);
- We provide an energy-aware solution to the placement problem (§3.2); and
- We carry out extensive experiments using the topologies of 4 real-world IoT deployments from different domains, comparing our approach to the state-of-the-art (§4.5).

2 Related work

2.1 The Need for Mediation in IoT systems

A fundamental challenge in designing IoT systems is to choose a communication protocol to be used by all device types regardless of function (sensing, actuating, processing, etc.), manufacturer, or computational capability [14]. A number of protocols have been proposed to enable such D2D communication. A prominent solution is the OASIS standard MQTT [7]: a simple and lightweight protocol that adopts a publish-subscribe paradigm and runs on top of TCP. MQTT defines a small message header, making it preferable for resource constrained networks. An alternative proposed by the IETF is CoAP [32], which follows a client-server paradigm, is based on the Representational State Transfer (REST) architecture, and runs on top of UDP. Other solutions include HTTP, AMQP [3], XMPP [2], among others.

Despite these attempts to standardize communication protocols, different IoT vendors still use varying messaging protocols [14, 27], which hampers IoT engineers from building more complex systems (*e.g.,* [16, 28]). As such, mediation between devices of different vendors is a common approach. Additionally, IoT systems designed by different teams of engineers are likely to use different protocols. To resolve this, mediation is typically used to act as a bridge between different protocols. For instance, the DeX framework [9] is a recent contribution to support mediation between different IoT protocols. However, little work has been done on how to optimize the placement of mediators in the network considering network and application constraints. A recent proposal [12], which we use as a baseline in our experiments, aims to do this while optimizing for delay-sensitive applications.

2.2 Virtual Network Function Placement

A related research topic is the placement of Virtual Network Functions (VNF) in order to optimize for certain objectives while meeting the system's functional requirements. Although the problem is similar at a high level, the solutions proposed in the literature (*e.g.,* [4, 11, 36, 37]) are not suitable as they optimize placement for different objectives such as link utilization and the size of the network forwarding table. A recent example [13] that is more pertinent to our problem presents an ILP-based model for the placement of virtual security functions (VSFs). The model considers server CPU capacities, VSF processing requirements, and network link capacities to calculate the optimal placement for minimizing energy consumption.

2.3 Energy-aware IoT

Optimizing energy consumption has been a long sought after goal in IoT systems. This problem has been tackled from different perspectives, such as switching to low-power communication technologies (*e.g.,* [29,35]), being selective about what data to aggregate/process/drop and where (*e.g.,* [5, 18, 22]), forecasting overall energy consumption [19], and so on.

Some proposals attempt to minimize the energy consumption of application servers within an IoT system (*e.g.,* [6]) and, as such, optimize for application metrics such as request satisfaction. However, none has tackled the challenge taking into consideration where to place mediation services and how this affects the energy consumption of D2D communication.

3 Energy-aware Placement

In this section, we present the system model and formulate the energy-aware mediator placement problem as an integer linear programming problem.

3.1 System Model

We consider an IoT system with a set of things $T = \{t_1, t_2, \ldots, t_m\}$, a set of access points $AP = \{ap_1, ap_2, \ldots, ap_k\}$, a set of nodes $N = \{n_1, n_2, \ldots, n_p\}$, and a gateway GW. The set of things includes sensors that read environmental data, actuators that effect actions, and external equipment that can be integrated into the network (*e.g.,* rescue teams equipment). The set of nodes consists of static machines that host mediation services to enable heterogeneous things to interact. The access points are hubs that connect the things and nodes to the gateway. In normal cases, the gateway connects the IoT system to the cloud where communication between devices occurs. However, when direct communication is required, which is the focus of this paper, the communication between things and nodes is always directed through the Gateway. Fig. 2 shows an typical topology of where things and nodes are located.

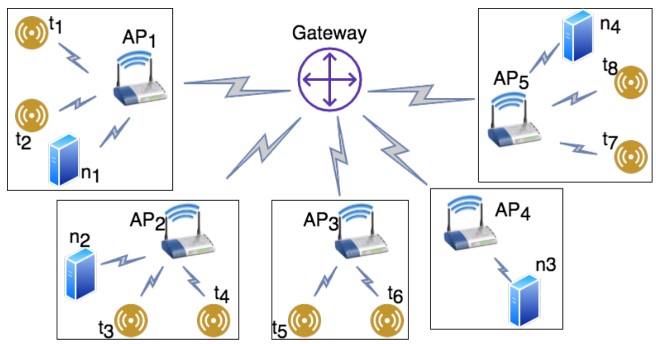

Fig. 2: Typical topology of an IoT system.

We assume that a location attribute $l = \{x, y, z\}$ is associated with each of the things, nodes, access points and the gateway where x, y and z are the coordinates of the location. We also assume that each of the things has a protocol attribute $p(t_i)$ that specifies the messaging protocol that defines the rules and formats of the messages exchanged with other things. The communication between the system things is represented as a set of interactions that occur during the lifetime of the system. We denote an interaction as i_j^{ab} where an interaction j involves things a and b. Each interaction involves sending messages of size $m(i_j)$ for a number of times $f(i_j)$.

The system is represented as a weighted graph $G = (V, E)$ where V and E denote sets of vertices and edges, respectively. Each vertex $v_i \in V$ represents a thing, node, access point or gateway. Each edge $e_{ij} \in E$ represents a link between two vertices and has a weight w_{ij} that indicates the available link bandwidth.

3.2 Problem formulation

The energy-aware mediator placement problem can be formally stated as follows: *Given a set of things, nodes, interactions and links, deploy the mediation service on a node so that the total energy consumed by the interactions is minimized provided that the bandwidth consumed on each link is constrained by the link's available bandwidth.* In the following we present how the end-to-end energy consumption is calculated and develop the objective function and constraints.

Links. In order to calculate the energy consumption of an interaction, we need to consider the energy consumed for transmitting and receiving data on each link that connects each pair of devices (thing, node, access point or gateway) along the interaction path. Consider the example given in Fig. 2. For an interaction that involves t_1 and t_5 where the mediation service is deployed on n_1, data will traverse the links $t_1 \rightarrow AP_1$, $AP_1 \rightarrow n_1$, $n_1 \rightarrow AP_1$, $AP_1 \rightarrow GW$, $GW_1 \rightarrow AP_3$, $AP_3 \rightarrow t_5$ (notation: $sender \rightarrow receiver$). On the link $t_1 \rightarrow AP_1$, energy consumed at t_1 to send the interaction messages from t_1 to AP_1 and energy is also consumed at AP_1 to received those messages; and so on for the other links. These links are grouped into a first-leg group of the interaction and a second-leg group where the first-leg includes links from the sending thing to the node hosting the mediation service and the second-leg includes links from the hosting node to the receiving thing. This grouping is important because the messaging protocol (and hence the message size) is different in the two legs. On the first-leg the used messaging protocol is the messaging protocol used by the sending thing ($p(t_1)$ in the above example) and on the second-leg the used messaging protocol is that of the receiving device ($p(t_5)$ in the above example).

Energy consumed per interaction. In order to calculate the consumed energy, we denote $\epsilon^T(d_u)$ and $\epsilon^R(d_u)$ for each device in the system, where the former refers to the transmission energy per bit and the latter refers to the receiving energy per bit of device d_u. Note that each of the transmitting and receiving devices can be a thing, node, access point or gateway. Now, in order to calculate the energy consumed for an interaction i_j^{ab}, we calculate the energy consumed by each leg of the interaction using equations 1, 2, and 3 where $m(i_j(p(t_a)))$ and $m(i_j(p(t_b)))$ are the message sizes of the messaging protocol of the sender and receiver things, respectively, and $\epsilon(i_j^{ab})$ is the energy consumed by the interaction.

$$\epsilon(first-leg_j) = (\sum(\epsilon^T(d_u)) + \sum(\epsilon^R(d_v))) \times m(i_j(p(t_a))) \times f(i_j) \qquad (1)$$

$$\epsilon(second-leg_j) = (\sum(\epsilon^T(d_u)) + \sum(\epsilon^R(d_v))) \times m(i_j(p(t_b))) \times f(i_j) \qquad (2)$$

$$\epsilon(i_j^{ab}) = \epsilon(first-leg_j) + \epsilon(second-leg_j) \qquad (3)$$

Objective function. Next, we calculate the total energy that is consumed by all the interactions so that we utilize it to reason about the selection of a node to host the mediator. Given a number of interactions n that occur in the system, the total consumed energy is calculated using equation 4 which sums up the energy consumed by each interaction.

$$\epsilon_{total} = \sum_{r=1}^{n} \epsilon(i_r^{ab}) \qquad (4)$$

Note that we do not include the processing energy consumption of the generation of the mediator nor the mediation because we assume these will be the same regardless of the where the mediation service is deployed.

Mediator host selection constraints. Given an interaction, the following two constraints must be satisfied for any mediation deployment to be acceptable:

- *Bandwidth constraint.* Given a host node and an interaction i_j^{ab}, the interaction will consume $m(i_j) \times f(i_j)$ bandwidth on every link along the interaction path from the source thing a to the destination thing b. The consumed bandwidth must be less than or equal to the available bandwidth on each of the links along the path. Algorithm 1 describes how this constraint is checked: it takes as input a potential hosting node, a graph representing the topology and lists of things and interactions. It returns $True$ if the bandwidth constraint is satisfied. It starts by extracting all the edges. Then, for each link, the algorithm accumulates the bandwidth that would be consumed by each interaction and checks if the total is less than the link bandwidth.

Algorithm 1 Check Bandwidth constraint

Input: A list of Things T, hosting node n_p, Interactions I, Graph G
Output: True: if the consumed bandwidth is less then the available, False: otherwise
For each edge e_i in G
  For each interaction i_j^{ab} in I
    Find a path leg_1 from t_a to n_p using the Breadth First search
    Find a path leg_2 from n_p to t_b using the Breadth First search
    For link (edge) el_m in $leg_1 \cup leg_2$
      If $el_m == e_i$
        $bandwidthUsed \mathrel{+}= m(i_j) \times f(i_j)$
      EndIf
    EndFor
      If $bandwidthUsed > bandwidth(e_i)$
        return False
      EndIf
  EndFor
EndFor
return $True$

- *Allocation constraint.* For each interaction i_j^{ab}, there is a set of nodes N that can host the mediation service of that interaction. However, for each interaction i_j^{ab}, we should only select one node to host the mediation service. We denote the selection of a node n_j to host the mediation service y_j^{ab}, the following constraint must be satisfied:

$$\sum_{n \in N} y_j^{ab} = 1 \tag{5}$$

Thus, the mediator placement problem is formalized with $w_{n_p}^{ab}$ being the bandwidth of the path from t_a to t_b when the mediator is hosted on node n_p:

$$\begin{aligned} \text{minimize} \quad & \epsilon_{total} \\ \text{subject to} \quad & \forall i_j^{ab} \;\; \forall n_p \sum m(i_j) \times f(i_j) \leq w_{n_p}^{ab} \\ & \sum_{n \in N} y_j^{ab} = 1 \end{aligned} \tag{6}$$

Table 1: A summary of the IoT deployments used for evaluation.

Name	#Things	#Nodes	Ref.	Use
AirPollution	14	6	[8]	Monitor city-wide air quality
SmartSantander	1,570	23	[30]	Monitor issues like noise, ambient temperature, light intensity, vehicle activity, CO levels, etc.
Sphere	1,500	500	[17]	Healthcare provision in residential environments
MassiveIrrigation	15,000	1,000	[23]	Manage freshwater distribution for precision irrigation of agricultural crops

4 Evaluation

To assess the efficiency and efficacy of our energy-aware placement algorithm, we run a set of rigorous experiments of mediator service placement in various contexts based on real-world IoT scenarios (§4.2). We compare our algorithm against three baselines: a naïve algorithm, a state-of-the-art one for delay optimization [12] (§4.3) and a state-of-the-art one for bandwidth optimization. We inspect the ability to improve different system performance metrics and the associated overhead (§4.4).

4.1 Experimental Setup

The experiments are conducted on a PC with Intel Pentium D 3.0GHz, 1GB RAM, running Linux Ubuntu v18. We used Java SE v1.8.0 to implement the placement algorithms and simulate the IoT infrastructure. We generate the parameters values as follows:

- Interactions are generated by randomly selecting two different things provided they have different messaging protocols so that a mediation service is required. The size and frequency of messages are generated randomly from the ranges [0,100] and [0,1000] respectively.
- Interface bandwidths of things, access points, and gateways are generated from the ranges [11,54], [11,54], [54,450] Mbps respectively according to the specification in [21].
- The values of energy per bit transmitted/received are generated from the ranges [5,20] mJ/bit [34] and [13.97,1902.11] nJ/bit [20].
- The locations of the system elements are generated within the Euclidean space of range [0,0] − [1000,1000] in meters.

4.2 IoT Contexts

We use 4 real-world IoT deployments as evaluation contexts. These were chosen to represent different scales and structures of IoT systems, as summarized in Table 1.

4.3 Baseline Placement Algorithms

- **Random** – In this naïve algorithm, a node to host the mediation service is selected at random from the list of potential hosting nodes, excluding those that violate bandwidth / allocation constraints.
- **Delay-optimized** – The algorithm proposed in [12] is, as discussed, the only contribution so far to address the mediation placement problem in IoT systems. The algorithm defines an objective function that aims to find a placement that minimizes the delay between interacting things. Delay is calculated as the sum of the transmission and propagation delays. The algorithm uses the absolute locations of things and nodes in the deployment environment to compute the propagation delay as the distance that data travel divided by the wave propagation speed. In other words, the algorithm makes no attempt to consider the network topology. In order to make a fair comparison with this algorithm, we modify the way distance is calculated to include the total distance between the sending thing and the receiving thing through the hosting node, access points and the gateway.
- **Bandwidth-optimized** – This algorithm determines placement such that the overall bandwidth consumed by D2D interactions is minimized. The algorithm calculates the bandwidth that interactions will consume on every link along the interaction path. It then sums up all the estimated bandwidth consumption on each link for each placement and solves the objective function to find the optimal placement.

4.4 Evaluation Criteria

The three algorithms are compared in terms of the following criteria:

- **Energy consumption** – The total energy consumed to deliver messages between things. We focus on transmitting and receiving messages, and ignore the energy of mediation assuming the latter is the same on all nodes.
- **Delay** – The end-to-end time delivery time between sender and receiver.
- **Execution time** – The time taken by the algorithm to find a placement of the mediation service.

4.5 Results

We presents our findings and draw comparison between the four algorithms.

Energy consumption The average values of energy consumption per interaction are depicted in Fig. 3. The plots indicate that significant amounts of energy could be saved using our placement algorithm. This per-interaction improvement ranges between 12.9% in the case of a small topology like the air pollution scenario, to 31.6% for large deployments such as the massive irrigation one. Energy consumption for the other placement algorithms is, overall, not better than

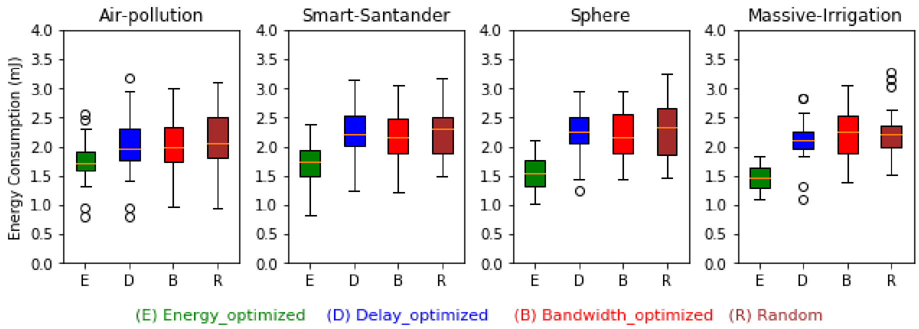

Fig. 3: The average energy consumed on message sending between source and destination, including intermediaries, at 250 interactions.

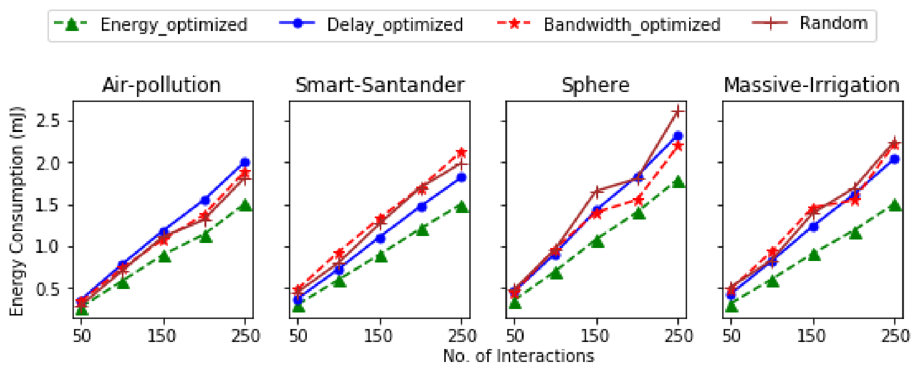

Fig. 4: The average energy of messaging for varied scales of interaction.

the random placement strategy. To further determine the scalability of the algorithms, we plot the energy consumption versus the number of interactions in Fig. 4. Our algorithm improves energy consumption for different levels of interaction. In addition, energy consumption grows as the number of interactions grow, which is due to demand for more traffic. Energy consumption increases linearly with the number of interactions, but with a steeper slope for all but our algorithm.

Delay Fig. 5 exhibits the average end-to-end messaging delay for each topology. The delay-optimized algorithm clearly achieves lower levels of delay than the alternatives. The amount of delay reduction is in the order of 3% in the case of small topology to 30.6% in the case of large topology – compared to the energy-optimized algorithm. Fig. 6 plots the delay at different interaction intensities. Unsurprisingly, the delay-optimized algorithm improves the delay for varied number of interactions. The effect of the scale of the topology is also evi-

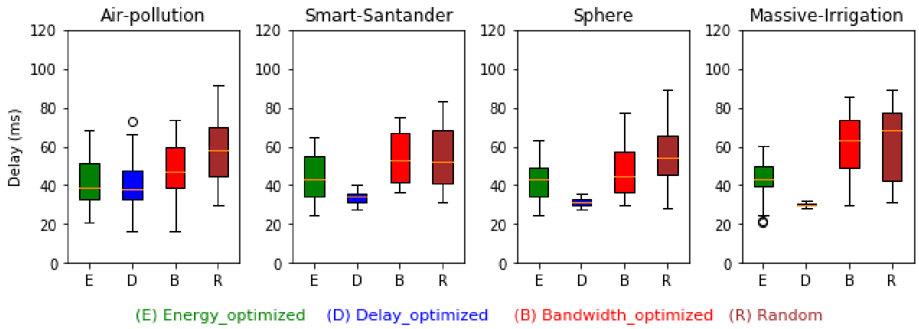

Fig. 5: The average end-to-end delay of message exchange at 250 interactions.

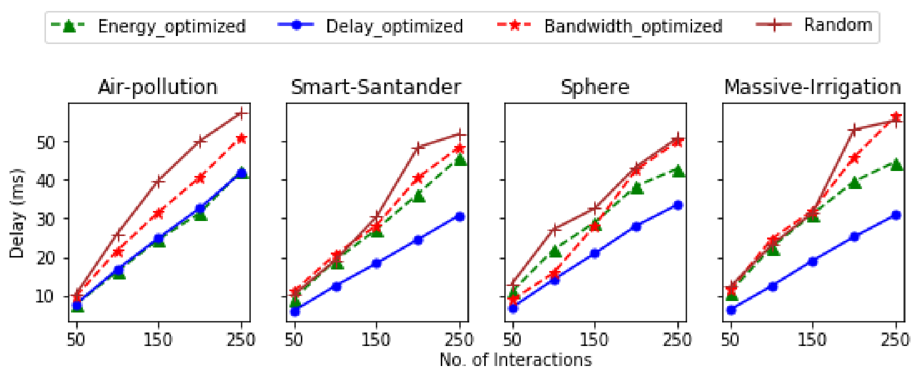

Fig. 6: The average end-to-end delay of message exchange for varied number of interactions.

dent as the slope of the linear relationship between increased traffic and delay: the larger the topology, the longer the delays.

Execution Time The plots in Fig. 7 portray the overhead in terms of execution time of each placement algorithm. All three non-trivial algorithms require very equivalent execution times. This is due to their similar levels of complexity, as all their run times scale with the number of device interfaces and interactions involved in the deployment. The last strategy requires the least due to it being a naïve one. With respect to scalability (Fig. 8), a linear trend with the increase of the number of interactions is again observed for all algorithms. This indicates that the energy-optimized algorithm is able to find the energy-optimal placement in a practicably acceptable runtime.

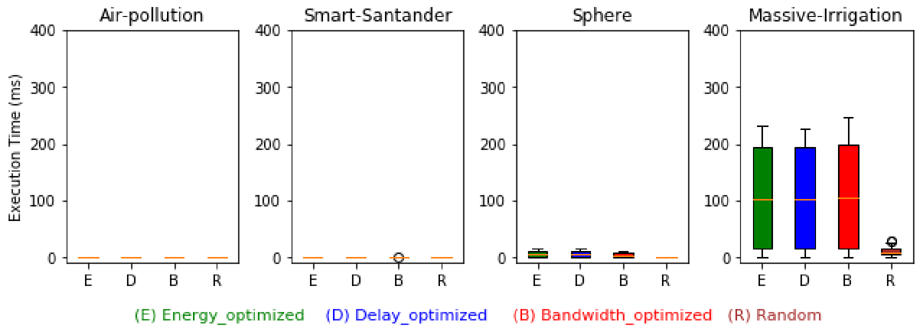

Fig. 7: The execution times to find the optimal placement at 250 interactions.

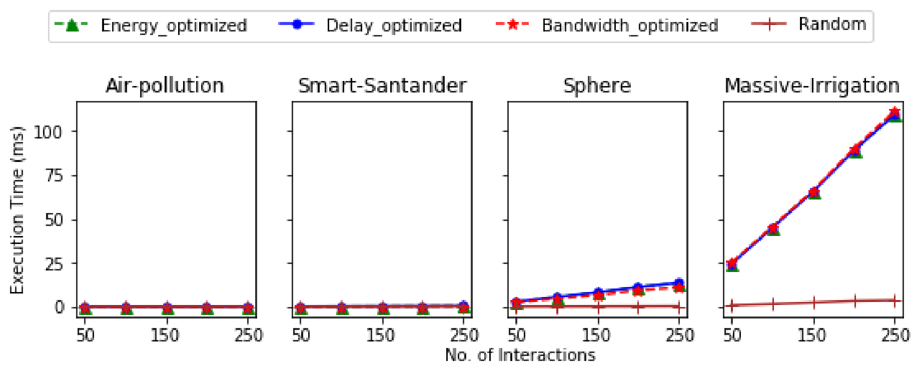

Fig. 8: The overhead in terms of average time taken by each algorithm to find the optimal placement.

5 Discussion

We now reflect on the implications of our findings, and lay groundwork for future work.

Trade-off and limitation – There is a clear advantage in terms of energy consumption at the expense of modest algorithm execution times and reversion to average messaging delays. In terms of making IoT deployments more sustainable and long-living, the latter overheads are deemed acceptable especially for large IoT deployments. The obvious limitation is that our approach is geared towards reducing energy consumption and not other metrics such as end-to-end delay. We aim to address this in future work (see point below).

Multi-objective optimization – The results presented in the previous section show that our proposed approach achieves lower energy consumption, but sometimes at the expense of higher end-to-end delay. Future work could build on both this and the delay-optimize alternative by defining the placement problem as a multi-objective optimization problem. Additionally, this can be extended by including other non-functional service objectives such as load balance, reliability, etc. Simultaneous optimization of multiple objective functions would require defining weights for each of the objectives of interest.

Adaptive placement – The inherent dynamism of IoT environments, arising from different factors (such as node mobility, usage patterns, failures, ephemeral nature), make adaptive placement a crucial operational procedure. One of the advantages of the presented approach is its reactive quality, through recalculation of the objective function. This adaptive capability can be further enhanced to provide proactive adaptation by utilizing techniques for change prediction.

Practicability – In the design of our optimization algorithm, there is an assumption that the scale of interactions between devices, and the volume of exchanged traffic is known beforehand. This is an unreasonable assumption for most real deployments. Instead, interaction frequency and volume could be estimated by analyzing historical data. This issue is similar to that of workload estimation in the cloud, (*e.g.,* [10])a field that can inform interaction estimation.

6 Conclusion

We propose an approach for placement of mediation services in an IoT system. The approach targets environments where IoT devices need to directly interact to exchange data. The approach is based on two key ideas. First, we formulate the placement problem as an integer linear programming problem taking into account the topology of the infrastructure. The proposed algorithm takes into consideration the energy consumed by each interaction along the path between source and destination things. Second, the approach devices an adaptive placement of the mediation services whereby recalculating the placement based on environmental changes. We demonstrate the feasibility of our approach through a methodology of quantitative evaluation, comparing our approach to base-lines from the literature. The results show that our approach provides a systematic way of finding a placement that minimizes energy consumption with a nominal computational overhead. This novel contribution has strong implications in IoT and CPS environments with direct device-to-device interactions and where minimizing energy consumption is needed for sustainable deployments.

Acknowledgments

This work was partially supported by the Adaptive Brokerage for the Cloud (ABC) project, UK EPSRC grant EP/R010889/1.

References

1. Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2030. https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/, accessed: 2021-04-08
2. Extensible messaging and presence protocol (XMPP): Core. RFC 3920, IETF (2004), https://tools.ietf.org/html/rfc3920
3. Advanced message queuing protocol version 1.0. Standard, OASIS (2012), http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-overview-v1.0-os.html
4. Afrin, M., Jin, J., Rahman, A.: Energy-delay co-optimization of resource allocation for robotic services in cloudlet infrastructure. In: Service-Oriented Computing. pp. 295–303. Springer (2018)
5. Azar, J., Makhoul, A., Barhamgi, M., Couturier, R.: An energy efficient IoT data compression approach for edge machine learning. Future Generation Computer Systems **96**, 168–175 (2019). https://doi.org/10.1016/j.future.2019.02.005
6. Badri, H., Bahreini, T., Grosu, D., Yang, K.: Energy-aware application placement in mobile edge computing: A stochastic optimization approach. IEEE Transactions on Parallel and Distributed Systems **31**(4), 909–922 (2020). https://doi.org/10.1109/TPDS.2019.2950937
7. Banks, A., Gupta, R.: MQTT version 3.1.1. Standard, OASIS (2014), http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html
8. Basford, P.J., Bulot, F.M.J., Apetroaie-Cristea, M., Cox, S.J., Ossont, S.J.: LoRaWAN for smart city IoT deployments: A long term evaluation. Sensors **20**(3) (2020). https://doi.org/10.3390/s20030648
9. Bouloukakis, G., Georgantas, N., Ntumba, P., Issarny, V.: Automated synthesis of mediators for middleware-layer protocol interoperability in the IoT. Future Generation Computer Systems **101**, 1271–1294 (2019). https://doi.org/10.1016/j.future.2019.05.064
10. Calzarossa, M.C., Massari, L., Tessera, D.: Workload characterization: A survey revisited. ACM Comput. Surv. **48**(3) (Feb 2016). https://doi.org/10.1145/2856127
11. Chen, X.: Energy efficient nfv resource allocation in edge computing environment. In: International Conference on Computing, Networking and Communications (ICNC). pp. 477–481 (2020). https://doi.org/10.1109/ICNC47757.2020.9049765
12. Chio, A., Bouloukakis, G., Hsu, C.H., Mehrotra, S., Venkatasubramanian, N.: Adaptive mediation for data exchange in IoT systems. In: Workshop on Adaptive and Reflexive Middleware. p. 1–6. ACM. https://doi.org/10.1145/3366612.3368122
13. Demirci, S., Sagiroglu, S., Demirci, M.: Energy-efficient virtual security function placement in nfv-enabled networks. Sustainable Computing: Informatics and Systems **30**, 100494 (2021). https://doi.org/10.1016/j.suscom.2020.100494
14. Dizdarević, J., Carpio, F., Jukan, A., Masip-Bruin, X.: A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration. ACM Comput. Surv. **51**(6) (2019). https://doi.org/10.1145/3292674
15. Elhabbash, A., Blair, G.S., Tyson, G., Elkhatib, Y.: Adaptive service deployment using in-network mediation. In: International Conference on Network and Service Management (CNSM). pp. 170–176 (Nov 2018)
16. Elhabbash, A., Nundloll, V., Elkhatib, Y., Blair, G.S., Sanz Marco, V.: An ontological architecture for principled and automated system of systems composition. In: International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS) (Jul 2020). https://doi.org/10.1145/3387939.3391602

17. Fafoutis, X., Elsts, A., Piechocki, R., Craddock, I.: Experiences and lessons learned from making IoT sensing platforms for large-scale deployments. IEEE Access **6**, 3140–3148 (2018). https://doi.org/10.1109/ACCESS.2017.2787418
18. Fitzgerald, E., Pióro, M., Tomaszwski, A.: Energy-optimal data aggregation and dissemination for the internet of things. IEEE Internet of Things Journal **5**(2), 955–969 (2018). https://doi.org/10.1109/JIOT.2018.2803792
19. Han, T., Muhammad, K., Hussain, T., Lloret, J., Baik, S.W.: An efficient deep learning framework for intelligent energy management in IoT networks. IEEE Internet of Things Journal **8**(5), 3170–3179 (2021). https://doi.org/10.1109/JIOT.2020.3013306
20. Hoque, M.A., Siekkinen, M., Nurminen, J.K.: Energy efficient multimedia streaming to mobile devices – a survey. IEEE Communications Surveys & Tutorials **16**(1), 579–597 (2014). https://doi.org/10.1109/SURV.2012.111412.00051
21. Intel: Different Wi-Fi protocols and data rates. `https://www.intel.com/content/www/us/en/support/articles/000005725/wireless/legacy-intel-wireless-products.html`, accessed: 2021-04-30
22. Jalali, F., Khodadustan, S., Gray, C., Hinton, K., Suits, F.: Greening IoT with fog: A survey. In: International Conference on Edge Computing. pp. 25–31 (2017). https://doi.org/10.1109/IEEE.EDGE.2017.13
23. Kamienski, C., Soininen, J.P., Taumberger, M., Dantas, R., Toscano, A., Salmon Cinotti, T., Filev Maia, R., Torre Neto, A.: Smart water management platform: IoT-based precision irrigation for agriculture. Sensors **19**(2) (2019). https://doi.org/10.3390/s19020276
24. Kim, T., Ramos, C., Mohammed, S.: Smart city and IoT. Future Generation Computer Systems **76**, 159–162 (2017). https://doi.org/10.1016/j.future.2017.03.034
25. Minoli, D., Sohraby, K., Occhiogrosso, B.: IoT considerations, requirements, and architectures for smart buildings—energy optimization and next-generation building management systems. IEEE Internet of Things Journal **4**(1), 269–283 (2017). https://doi.org/10.1109/JIOT.2017.2647881
26. Montori, F., Bedogni, L., Bononi, L.: A collaborative internet of things architecture for smart cities and environmental monitoring. IEEE Internet of Things Journal **5**(2), 592–605 (2018). https://doi.org/10.1109/JIOT.2017.2720855
27. Naik, N.: Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: International Systems Engineering Symposium. pp. 1–7 (2017). https://doi.org/10.1109/SysEng.2017.8088251
28. Nundloll, V., Elkhatib, Y., Elhabbash, A., Blair, G.S.: An ontological framework for opportunistic composition of IoT systems. In: International Conference on Informatics, IoT, and Enabling Technologies (ICIoT). pp. 614–621. IEEE (Feb 2020)
29. Qin, H., Cao, B., He, J., Xiao, X., Chen, W., Peng, Y.: Cross-interface scheduling toward energy-efficient device-to-gateway communications in IoT. IEEE Internet of Things Journal **7**(3), 2247–2262 (2020). https://doi.org/10.1109/JIOT.2019.2958612
30. Sanchez, L., noz, L.M., Galache, J.A., Sotres, P., Santana, J.R., Gutierrez, V., Ramdhany, R., Gluhak, A., Krco, S., Theodoridis, E., Pfisterer, D.: SmartSantander: IoT experimentation over a smart city testbed. Computer Networks **61**, 217–238 (2014). https://doi.org/10.1016/j.bjp.2013.12.020, special issue on Future Internet Testbeds – Part I
31. Serrano, M., Gyrard, A., Tragos, E., Nguyen, H.: FIESTAIoT Project: Federated Interoperable Semantic IoT/Cloud Testbeds and Applications. In: The Web Conference. p. 425–426 (2018). https://doi.org/10.1145/3184558.3186199

32. Shelby, Z., Hartke, K., Bormann, C.: The constrained application protocol (CoAP). RFC 7252, IETF (2014), https://www.rfc-editor.org/rfc/rfc7252.txt
33. Sinche, S., Raposo, D., Armando, N., Rodrigues, A., Boavida, F., Pereira, V., Silva, J.S.: A survey of IoT management protocols and frameworks. IEEE Communications Surveys & Tutorials **22**(2), 1168–1190 (2020). https://doi.org/10.1109/COMST.2019.2943087
34. Sun, L., Sheshadri, R.K., Zheng, W., Koutsonikolas, D.: Modeling WiFi active power/energy consumption in smartphones. In: International Conference on Distributed Computing Systems. pp. 41–51 (2014). https://doi.org/10.1109/ICDCS.2014.13
35. Wilhelmsson, L.R., Lopez, M.M., Sundman, D.: NB-WiFi: IEEE 802.11 and Bluetooth low energy combined for efficient support of IoT. In: Wireless Communications and Networking Conference. pp. 1–6 (2017). https://doi.org/10.1109/WCNC.2017.7925808
36. Xu, Z., Zhang, X., Yu, S., Zhang, J.: Energy-efficient virtual network function placement in telecom networks. In: International Conference on Communications (2018). https://doi.org/10.1109/ICC.2018.8422879
37. Zong, Y., Ou, Y., Hammad, A., Kondepu, K., Nejabati, R., Simeonidou, D., Liu, Y., Guo, L.: Location-aware energy efficient virtual network embedding in software-defined optical data center networks. J. Opt. Commun. Netw. **10**(7), 58–70 (2018). https://doi.org/10.1364/JOCN.10.000B58