Scheduling Services on an IoT Device Under Time-Weighted Pricing

Ioannis Avgouleas, Nikolaos Pappas and Vangelis Angelakis

Scheduling Services on an IoT Device under Time-Weighted Pricing

Ioannis Avgouleas, Nikolaos Pappas, and Vangelis Angelakis

Department of Science and Technology,
Linköping University, Campus Norrköping, 60 174, Sweden
e-mails: {ioannis.avgouleas, nikolaos.pappas, vangelis.angelakis}@liu.se

***Abstract*—The emerging vision of smart cities necessitates the use of Internet of Things (IoT) network devices to implement sustainable solutions that will improve the operations of urban areas. A massive amount of smart cities services may demand allocation of computational resources, such as processing power or storage, that IoT devices offer. Within this context, we present an IoT network device comprising interfaces with one specific computational resource available. The efficient utilization of available IoT resources would improve the Quality of Service (QoS) of the IoT network that serves the smart city. All resource allocations must be completed within a given scheduling window and every service is parametrized by a pricing weight function to indicate its tolerance to be served at the beginning of the scheduling window. We propose a mathematical optimization formulation to minimize the total cost of allocating all demands within the scheduling window considering the tolerance level of each service at the same time. Moreover, we prove that the problem is computationally hard and we provide numerical results to gain insight into the impact of different pricing weight functions on the allocations' distribution within the scheduling window.**

I. Introduction

Undoubtedly the majority of people live in cities and the urban population is expected to grow up to five billion by 2030 [1], namely 60% of the world population will be living in cities by that time. The vision of smart cities embraces developments of sustainable solutions to improve the operations of urban areas. Emerging Information and Communication Technologies (ICT) solutions will provide the necessary network infrastructures upon which that vision is going to be built [2]. Among the available ICT solutions, the evolution of the Internet of Things (IoT) is one of the most promising ones [3].

The IoT comprises devices with computing, processing, storing, sensing, and communication capabilities. Their interconnection allows for constant exchange of information between their smart sensing environment and the Internet world. The IoT paradigm has many applications in numerous domains relevant to the smart cities vision such as home and industrial automation, waste management, health-care monitoring and assistance, elderly supervision, energy management in smart grids, and traffic management, just to name a few [4, 5].

In recent years, IoT and smart cities converge more and more due to national policies towards implementing ICT services to administer urban areas and exploit economies of scale through the development of sustainable solutions. The potential of connecting urban objects and services to the Internet to enable their remote management and utilization can have a beneficial impact on both the residents and the authorities. Such an envisioned connected city can offer improved everyday life quality to the citizens at lower operational costs for the municipal authorities. Moreover, from the IoT perspective, smart cities offer challenging yet potentially highly lucrative application scenarios. As a result, many IoT technological advancements have been motivated, tested, and implemented solely for such scenarios.

Within this context, in this paper, we assume an IoT network device consisting of interfaces offering one specific computational resource, such as processing power or storage, for smart cities services. Given the massive demands of smart cities services, available resources may be scarce. Thus, an efficient usage of IoT resources would improve the QoS of the IoT network that serves the smart city. The scheduling, namely the allocation of resources over time, is done within a given period which we call scheduling window. Within the latter, all services' demands for resources must be served. We use pricing weight functions to model the fact that some services may demand to be fully served early in the scheduling window, whereas others may be indifferent regarding their serving time.

A. Contribution

We introduce a mathematical formulation that minimizes the total cost of allocating all demands of services to available resources within a given scheduling window. We enhance the formulation of [6] to include pricing

The authors would like to thank Professor Di Yuan for his help in this work. At the time of submission of this paper both V. Angelakis and I. Avgouleas were seconded at Converge ICT, within the EU FP7-IAPP project no. 612361 SOrBet.

weight functions that indicate each service's tolerance levels. Intolerant services weigh leftover allocations more at the beginning, while more tolerant services weigh leftovers more towards the end of the scheduling window. We call this the *Pricing-Weighted-Services Problem (PWSP)* and prove its NP-Completeness. Additionally, we provide numerical results to gain insight into the impact of the pricing weight functions on the allocations' distribution over time. Moreover, we show how services with different mixes of weight functions affect the finishing serving times, namely the last time-slot by which every service has been served.

II. Modeling *PWSP* as an Integer Program

We consider a set of interfaces $\mathcal{I} = \{1, \ldots, I\}$ that offer capacities of available computational resources of one type. Each service $j \in \mathcal{J} = \{1, \ldots, J\}$ demands a certain amount of this resource. The demands are modeled as integer parameters d_j. Likewise, each interface i offers capacity for this resource denoted as an integer parameter b_i. We also consider a scheduling window of fixed size T and discretize time into slots $t \in \mathcal{T} = \{1, 2, ..., T\}$.

In the model that follows, we use the integer decision variable x_{ijt} to model the resulting allocations i.e., the amount of the resource of the j-th service that is served by interface i at time-slot t. We denote by c_{ijt} the cost to serve one unit of resource on interface i for service j at time-slot t. The activation cost of the i-th interface at time t is F_{it}, and the binary decision variable A_{it} is one if and only if there is at least one resource served by interface i at time-slot t.

The mathematical formulation, which uses B to denote the sum of the interfaces' capacities for every time-slot, is the following:

$$[PWSP] \text{ min.} \quad \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{T}} c_{ijt}x_{ijt} + \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}} F_{it}A_{it} + \sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} w_{jt}l_{jt} \tag{1}$$

$$\text{s.t.} \quad \sum_{i\in I}\sum_{t\in\mathcal{T}} x_{ijt} = d_j,\ \forall j \in \mathcal{J}, \tag{2}$$

$$\sum_{j\in\mathcal{J}} x_{ijt} \le b_i,\ \forall i \in \mathcal{I},\ \forall t \in \mathcal{T}, \tag{3}$$

$$\sum_{j\in\mathcal{J}} x_{ijt} \le A_{it}B,\ \forall i \in \mathcal{I},\ \forall t \in \mathcal{T}, \tag{4}$$

$$\sum_{t\in\mathcal{T}} w_{jt}l_{jt} \le \theta_j,\ \forall j \in \mathcal{J}, \tag{5}$$

$$l_{jt} \triangleq \begin{cases} d_j - \sum_{i\in\mathcal{I}} x_{ijt}, & t = 1, \forall j \in \mathcal{J} \\ l_{j(t-1)} - \sum_{i\in\mathcal{I}} x_{ijt}, & 2 \le t \le T, \forall j \in \mathcal{J} \end{cases} \tag{6}$$

TABLE I
Parameters used in our Experiments

Pricing function	**Weights**
Constant	$w(t) = 1,\ \forall t \in \mathcal{T}$
Increasing	$w_+(t) = t,\ \forall t \in \mathcal{T}$
Decreasing	$w_-(t) = \lvert\mathcal{T}\rvert - t, \forall t \in \mathcal{T}$
Parameter	**Value**
Scheduling window size	$\lvert T\rvert = 4$ time-slots
Interface capacity	$b = 35, \forall t \in \mathcal{T}$
Services demands	$d = [45, 45, 45]$
Services budgets	$\theta = [220, 220, 220]$
Cost per unit allocation	$c_{1t} = 0, \forall t \in \mathcal{T}$
Activation cost	$F_{1t} = 0, \forall t \in \mathcal{T}$

$$x_{ijt} \ge 0\ integer,\ \forall i \in \mathcal{I},\ \forall j \in \mathcal{J},\ \forall t \in \mathcal{T}. \tag{7}$$

$$A_{it} \in \{0, 1\},\ \forall i \in \mathcal{I},\ \forall t \in \mathcal{T}, \tag{8}$$

$$l_{jt} \ge 0\ integer,\ \forall j \in \mathcal{J},\ \forall t \in \mathcal{T}, \tag{9}$$

The first constraint ensures that all services' demands are met. The second constraint guarantees that there are enough interfaces' capacities to serve the whole demands set. The third constraints makes sure that the activation variable, for an interface, at a specific time-slot, will be set to one if least one service occupies the interface at that time-slot. The fourth constraint ensures the leftovers, namely the unserved demands, are penalized according to the given pricing weight function w_{jt} for each service j at time-slot t. The parameter that specifies the upper bound on the aggregate weighted leftovers for each time-slot is called service budget and is denoted with θ_j for the j-th service. The fifth constraint defines the leftovers for each service at each time-slot, and the last three constraints provide the domains for the decision variables. The objective is to minimize the total cost of serving the whole demands set, namely the total resource utilization and activation cost as well as the total cost of the weighted leftovers. The *PWSP* is NP-hard. The proof is based on a standard reduction to the Capacitated Facility Location problem and is omitted due to space limitation.

III. Model Verification Results

In this section, we present simple numerical results to give insight into the system model we described in section II. More specifically, we will discuss how the service budgets of (5) in conjunction with the pricing weight functions affect the distributions of allocations over time.

We used three different pricing weight functions (see TABLE I) to demonstrate how they affect the finishing serving time i.e., the time-slot by which the demands of a service have been fully served. The pricing functions reflect how urgent the finishing serving time of a service is. For instance, services that are very intolerant – in terms of their finishing serving time – should use the

decreasing weight function w_- that penalizes more the leftovers at the beginning of the scheduling window. Depending on the available capacities, the bulk or even all demands of intolerant services may be served at the beginning of the scheduling window. Less intolerant services should use the increasing weight function w_+ that penalizes more the leftovers at the end of the scheduling window. Services using w_+ will get less priority over w_- at the beginning but will be prioritized later in the scheduling window. Tolerant services i.e., ones that are priced with the constant weight function w, will fill in any allocations gaps of each time-slot since w equally penalizes the leftovers of each time-slot.

In the following results, we consider an IoT device consisting of $|\mathcal{I}| = 1$ interface with the parameters of TABLE I. The problem is feasible since the whole demands set can be served by the interface's capacity. Moreover, the cost per unit of allocation, c_{ijt}, has been set to be constant and the same for every service to avoid interfering with the effect of $\theta's$ and w_{jt} on the allocations. Furthermore, the activation costs of the interfaces, F_{it}, have been set to zero for the same reason. Consequently, with this set of parameters, the allocations' distributions over time largely depend on the given pricing weight functions, the service budgets as well as the interface capacity.

In the plots of this section, we present results of scheduling three services using different sets of weight functions to see how each service is prioritized based on the pricing function of that service and the rest ones. The results in Fig. $1(a)$ show the allocations over time when two services use the constant weight function w and one uses the increasing one w_+. Every service has equal demands ($d_j = 45$), and the same service budget ($\theta_j = 220$). Service 1 is getting priority over the tolerant services, since its pricing function penalizes more its leftovers. Thus, it terminates in the second time-slot and no leftovers are charged for that time-slot. The tolerant services (i.e., 2 and 3) finish allocation after the intolerant service and the order by which this happens is indifferent since their weight function does not indicate any specific intolerance between them. On the other hand, in Fig. $1(b)$, service 1 occupies as much capacity as possible in the first time-slot to leave as few leftovers as possible, since w_- penalizes mostly leftovers in the beginning of the scheduling window.

In Fig. 2, a service with the constant weight function is replaced with a service with the increasing or the decreasing weight function. The intolerant services i.e., service 1 and 2 are getting priority over the tolerant service which finishes last occupying the whole time-slot. Additionally, the tolerant service starts being served after the bulk of the intolerant services' demands have been served. In Fig. 3, services with only increasing or only decreasing functions are scheduled. Thus, every service has the same demands, weight functions, and service budgets. As expected, no service is getting priority over the other since every parameter is equal. In Fig. $3(a)$, every service finishes allocation at the last time-slot since no demand can be entirely served in one time-slot and has to be split (see parameters in TABLE I). In Fig. $3(b)$, no service is prioritized as well. For example, the most served service in the first time-slot is service 1 but finishes allocation largely at the end, and service 2 is largely being served at time-slot 2 but finishes allocation at the last time-slot as well. This is done because with the specific parameters there are a lot of optimal solutions and the solver choses an optimal solution minimizes the objective function with the faster pace. Consequently, the plot depicts one of the possible optimal solutions.

IV. Results

We performed several sets of simulations to assess the allocations for configurations of 100 services using different sets of pricing weight functions to gain insight into how they affect the total cost. We use parameters such that *PWSP* is always feasible i.e., the whole set of demands can always be satisfied by the available capacities and the scheduling window has 4 time-slots. We use one IoT interface since we are interested in how the weight functions affect the distribution of the allocations rather than how the demands are split among different interfaces. We considered different runs in our simulation. Each run uses the same demand for each of the 100 services and the same cost per unit allocation. The mixture of the pricing weight functions of each run is different. Each mixture consists of constant, increasing, and decreasing pricing weight functions as per TABLE I. However, every mixture uses a different percentage of these three classes of weight functions. Mixture 1 consists of 100 services, out of which $\mathcal{N}(10, 1)$ use the increasing weight function, $\mathcal{N}(40, 3)$ use the decreasing weight function, and the rest are services with constant weight functions. Please note that we denote as $\mathcal{N}(\mu, \sigma^2)$ the Gaussian probability distribution function with mean μ and standard deviation σ. In mixture 2, out of 100 services, $\mathcal{N}(40, 3)$ of them use increasing weight functions, $\mathcal{N}(10, 1)$ use decreasing weight functions, and the rest use constant weight functions. Mixture 3 is similar to the first one, but we exchange the increasing function w_+ of TABLE I with a more intolerant increasing function: $w_{++} = 2w_+$. Hence, this mixture consists of $\mathcal{N}(10, 1)$ services utilizing the w_{++} weight function, $\mathcal{N}(40, 3)$ services utilizing the decreasing weight function, and the rest utilizing constant weight functions. We ran experiments for 1000 runs to gain insight into how different mixtures of the pricing weight functions affect the resulting allocations. The plots of Fig. 4 show the 95% confidence interval of allocations over time for

(a) (b)

Fig. 1. Scheduling 3 services: two of them use of the constant weight function w and the other one uses either (a) the increasing function w_+, or (b) the decreasing function w_-. In both cases, the two tolerant services finish after the intolerant one. In (b), the most intolerant service w_- occupies the whole capacity of the interface at the first time-slot and the reminiscent allocations take place in the second time-slot. If w_+ is used instead of w_-, as in (a), then the bulk of the intolerant service is served at the second time-slot. The pricing function penalizes the leftovers of the corresponding time-slot, and, hence, no cost is charged for the allocation of service 1 at time-slot 2 since there are no leftovers after this allocation.

(a) (b)

Fig. 2. Scheduling 3 services: one uses the constant weight function w and two use either (a) the increasing function w_+, or (b) the decreasing function w_-. The demands of the intolerant services (i.e., 1 and 2) occupy the first two time-slots and their finishing time is the third time-slot. The tolerant service starts serving after the tolerant services finishing time and the bulk of its allocation takes place at the last time-slot.

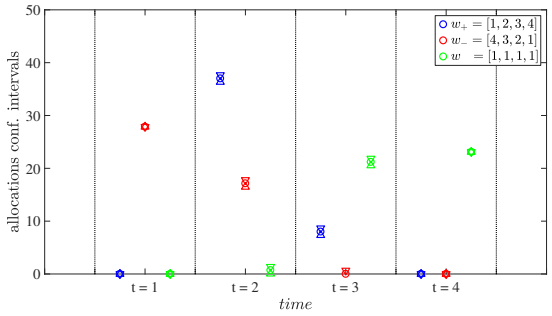
(a) (b)

Fig. 3. Scheduling 3 services: (a) only increasing, or (b) only decreasing functions. In (a), every service finishes allocation at the last time-slot, since no demand can be entirely served in one time-slot and has to be split over time. In (b), the solver choses one of the many directions that minimize the cost and the plot depicts the resulting scheduling. No service is really prioritized since every service parameter is the same.

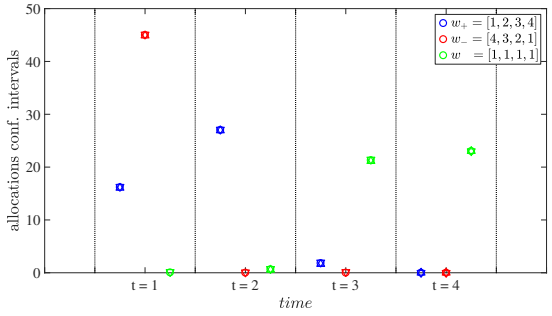
1000 runs for the three previously described mixtures of 100 services. Since every parameter - apart from the pricing weight functions - was the same for every run, the latter and the interface capacity affect the allocations mostly.

When the mixture comprises of more decreasing (and more intolerant) than increasing (and more tolerant) services, such as mixture 1 (see Fig. 4(a)), a large part of the intolerant services is served as early as possible within the scheduling window. If the interface capacities are enough, even the whole set of intolerant services can be served at the first time-slot. The next more intolerant class of services, that use weight function w_+, gets the remaining capacities that the intolerant class of services has left at the second time-slot of the scheduling window. Finally, the most tolerant services, that use the w weight function, start allocating largely at the end of the scheduling window when the other classes of services are already fully served.

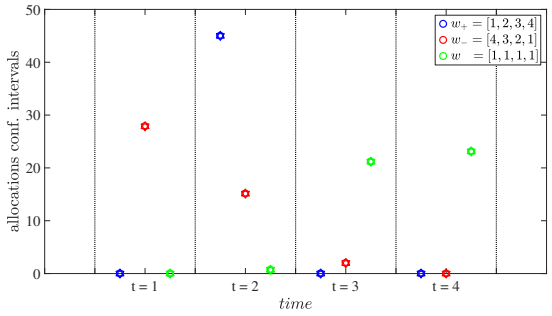
When the mixture comprises of more services with increasing functions rather than decreasing ones and the demands of the most intolerant class can be entirely served by the interface at the beginning of the time-slot, then the most intolerant class of services is fully



(a) Mixture 1 consists on average of 10% of services with the w_+ weight function, 40% of services with the w_- weight function, and the rest using w.



(b) Mixture 2 consists on average of 40% of services with the w_+ weight function, 10% of services with the w_- weight function, and the rest using w.



(c) Mixture 3 consists on average of 10% of services with the w_{++} weight function, 40% of services with the w_- weight function, and the rest using w.

Fig. 4. Allocations over time for mixtures of 100 services using different pricing weight functions. Mean values are denoted with o, while symbols ∇ and Δ indicate the upper and the lower bound of the corresponding 95% confidence interval, respectively. The plots show results of 1000 runs for: (a) Mixture 1, (b) Mixture 2, (c) Mixture 3.

served as soon as possible as in Fig. 4(b). Thus, services with weight w_- are fully served in the first time-slot and services with w_+ get allocated at the remaining capacity of that time-slot since they are more costly than services with w. Consequently, tolerant services start being served at the end of the scheduling window, namely mostly at the last two time-slots.

When the mixture consists of more aggressive increasing weights such as mixture 3, services with the increasing weight function are getting allocated as close to the beginning of the scheduling window as possible since, after the first time-slot, its increasing weight function penalizes leftovers even more than the decreasing one. Therefore, services with weight w_{++} are fully served by the second time-slot comparing to the third time-slot of Fig. 4(a) since the interface capacities of a single time-slot are enough to serve them (see in Fig. 4(c)).

V. Conclusion

This paper addresses the problem of allocating resources that smart cities' services demand from an IoT device. The model takes into account the tolerance level of each service to minimize the cost of allocating all demands on the device's network interfaces. The total cost consists of the cost of allocating every resource unit, the activation cost of each interface, and the cost that each service's pricing weight function incurs.

Several sets of simulations have been performed to assess how using different sets of pricing weight functions affects the allocations' distribution over time. If a mixture comprises of more intolerant than tolerant services, the former get priority and, thus, at least a large part of the former is served as early as possible. Conversely, if a mixture comprises of more tolerant than intolerant services and the demands of the latter can be entirely served at the beginning of the scheduling window, then the latter are fully served as soon as possible. Overall, the numerical results successfully indicate the ability of our formulation to model the tolerance levels of smart cities services.

References

[1] "Resilient people, Resilient Planet: A future worth choosing," *United Nations Secretary-Generals High-Level Panel on Global Sustainability*, 2012.

[2] V. Angelakis, E. Tragos, H. Pöhls, A. Kapovits, and A. Bassi in *Designing, Developing, and Facilitating Smart Cities*, Springer International Publishing, 2017.

[3] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," *Computer Communications*, vol. 54, pp. 1 – 31, 2014.

[4] S. P. Mohanty, U. Choppali, and E. Kougianos, "Everything you wanted to know about smart cities: The Internet of Things is the backbone," *IEEE Consumer Electronics Magazine*, vol. 5, pp. 60–70, July 2016.

[5] B. Ahlgren, M. Hidell, and E. C. H. Ngai, "Internet of things for smart cities: Interoperability and open data," *IEEE Internet Computing*, vol. 20, pp. 52–56, Nov 2016.

[6] V. Angelakis, I. Avgouleas, N. Pappas, E. Fitzgerald, and D. Yuan, "Allocation of heterogeneous resources of an iot device to flexible services," *IEEE Internet of Things Journal*, vol. 3, pp. 691–700, Oct 2016.