Minimal Wireless Broadcast Schedules for Multi-objective Pursuits

Christos Liaskos, *Member, IEEE,* Angeliki Tsioliaridou, Georgios Papadimitriou, *Senior Member, IEEE,* and Petros Nicopolitidis, *Senior Member, IEEE*

***Abstract*—Mobile wireless devices can now act as push-based data broadcasters, disseminating information in crowded places. While assuming this role, a device must take into account various factors, such as the mean service time, the energy expenditure or copyright costs. Apart from adhering to multi-objective optimality, a broadcast schedule must also be as small as possible in size, in order to be producible and cache-able on a mobile device. The present paper analytically proves the existence of optimal schedules that are finite, minimal in size. Furthermore, the optimality of the schedule is allowed to refer to an unlimited number of criteria. The analysis shows that multi-objective optimality can be met precisely by extremely small schedules, enabling their use in mobile devices. Simulations indicate perfect efficiency of the minimal schedules in realistic application scenarios.**

***Index Terms*—Data broadcasting, Push systems, Schedule size minimization, Multi-objective optimality.**

I. Introduction

As the number of networked Web users and entities grows worldwide, common information needs begin to manifest. Push-based broadcasting constitutes then a promising dissemination scheme, especially in wireless environments, since it can serve multiple users in one go [1]. Data push operates by streaming content to clients without their direct request, since their overall needs have been statistically modeled in advance. TV broadcasting and Web media channels are classic examples of push-based operation [2]. The push paradigm offers increased bandwidth efficiency and unlimited scalability [3]. The device that schedules the pushing of data to a set of clients is typically considered to be a powerful and expensive server, e.g. a TV broadcast station [4]. The need for powerful hardware is due to the typically high data volume to be pushed, and because of the need to take multiple criteria into account during the process. For example, the data may be associated to popularity metrics, copyright costs, media quality levels, or even content types and ratings. Thus, data push has been considered as an infrastructure-based solution only, implemented in e.g. cellular telephony and TV networks [5], [6].

However, a mobile, ubiquitous computing era has emerged. In many cases we can find tenths or hundredths of portable devices within the vicinity of our smartphone. Bringing data push to the "weak" smartphone hardware would allow for a new class of mass notification and advertisement applications. Examples include Point-of-Interest updates in crowded places, such as price alerts in malls, accident notifications in VANETs or even portable media streaming stations. The data volume for push can be kept at tractable levels, since it has local value. However, the need for taking multiple optimization objectives into account during the push scheduling process remains. The present work will analytically minimize the complexity of multi-optimal, data push scheduling. Thus, the analysis contributes the most lightweight version of push scheduling possible, enabling its implementation in current or future portable devices.

The study assumes the standard model for push systems [1]. A wireless broadcast server manages the push of a set of data items to a client set. Each item is characterized by attributes expressed in scalar form. Standard attributes are the statistical popularity and size of an item. Other attributes may include a copyright cost per broadcast of a data item, a content rating, e.t.c. [7]. The broadcast server produces a push schedule which dictates which item should be pushed at any given time. The performance of a schedule is characterized by the mean client waiting time it achieves [1], [8]. Push schedules are always periodic, i.e. repeating patterns, due to the absence of explicit client queries [9]. Each item has its own broadcast period, derived from its attributes. E.g. more popular items are pushed more often. It is worth noting that a schedule has typically a limited lifetime. The item attributes can vary, at which point the scheduling algorithm is simply re-run to produce an up-to-date schedule. The study will not considered such adaptivity pursuits, since they are completely modular to the presented scheme (simple re-run) and are examined as a separate field of study elsewhere [6]. However, given a lifetime, the running costs of a schedule become the definitive performance tuning factors [7]. For example, the mean (running) copyright cost over all scheduled broadcasts multiplied by the lifetime yields the total monetary cost of a push service. Nonetheless, setting the item broadcast periods primarily by copyright and not by popularity induces a performance trade-off, raising the mean client waiting time. Therefore, a multi-optimal schedule is defined as one the minimizes the mean waiting time of the clients, subject to multiple mean (running) cost restrictions [7], [9].

The present paper will contribute the analytical minimization of the computational complexity of any multi-optimal push schedule. The study builds upon the conclusion of [4], which showed that the complexity of a periodic schedule is proportional to its size, i.e. the total number of item broadcasts it includes. However, [4] i) provided heuristic indications of

C. Liaskos and A. Tsioliaridou are with the Foundation of Research and Technology - Hellas (FORTH), N. Plastira 100 Vassilika Vouton, GR-700 13 Heraklion, Crete, Greece, e-mail: [cliaskos, atsiolia]@ics.forth.gr.

G. Papadimitriou and P. Nicopolitidis are with the Department of Informatics, Aristotle University, P.O.B. 888, Thessaloniki 54124, Greece, e-mails: [gp, petros]@csd.auth.gr.

this claim only and ii) was limited to the minimization of the mean client waiting time, without any cost considerations. In the present study, we minimize analytically the size of any push schedule with multiple cost restrictions (Section III). The methodology is based on a novel analytical approach, which transforms the discrete item scheduling problem to an equivalent, tractable, continuous representation. Finally, the simulations conducted in Section IV demonstrate the immediate benefits of the minimal schedules in realistic application scenarios.

II. Related Work

Broadcast scheduling may be either an on-demand (pull-based) or a probabilistic (push-based) process. Both approaches target the minimization of the clients' mean serving time, or other similar criteria. However, the former operates on explicit client queries [10], [11], while the latter assumes only statistical knowledge of the popularity of the data items. The present work refers to push-based scheduling. Furthermore, the push scheduling process comprises two parts: the definition of optimal number of item occurrences in the schedule and the construction of the actual broadcast sequence. This work is related to the second part, while the first one is an NP-Hard problem [12].

Research on wireless broadcast systems, initially focused on the analytical minimization of the clients' mean serving time, in the context of Teletext systems [13]. It was proven that an optimal schedule is also periodic. Therefore, one needs to define only the optimal number of occurrences of each data item inside the schedule. Gecsei [13] provided a solution, assuming equally sized data items. The problem was revisited in [14], heuristically studying items with small variation in their sizes. It was clarified, that the mean serving time depends on data item attributes (i.e., item request probabilities and sizes), and not on the number of clients. The lower bound for the mean serving time was calculated in [1]. The same study proposed a water-filling-based scheduling algorithm that achieved optimality at the expense of increased complexity. It relied on a "golden" analytical condition that must be upheld during the operation of the system. At each broadcast decision, the algorithm strives to conform to this condition by equalizing the deviation over the data items. The algorithm achieved periodicity only statistically, over an infinite broadcast schedule. In general, strict periodic push of items with different broadcast periods leads to collisions, whose optimal resolution is a combinatorial, NP-Hard problem [12]. Liaskos et al. [4] showed that: i) preemptive broadcasting [5] alleviates the NP-Hardness. ii) Scheduling complexity is proportional to the size of the produced schedule. Simulation results then implied that optimality could be feasible with finite schedules.

Heuristic scheduling methods were also introduced in [15] with the introduction of the Broadcast Disks model. According to it, items are grouped by popularity, forming virtual disks rotating around a common axis. Imaginary heads read and serialize data from the disks, producing the final schedule. Further studies on the model have addressed the issues of multichannel data push [16], serving correlated items [17],

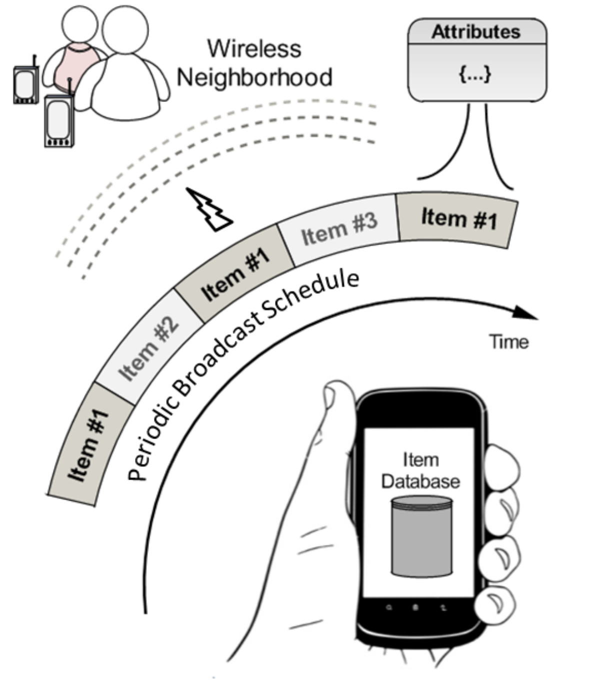


Figure 1. Overview of the system model. A portable device broadcasts data to its neighbor using a periodic schedule approach. Each item has any number of associated attributes, which may exemplary include item-specific copyright cost and battery life drain per broadcast.

pull-push hybrid data dissemination [18] and data indexing for saving power at the receiving devices by shutting down the network interface for prolonged intevals [19]–[21].

All these studies focused on the minimization of the clients' mean serving time, disregarding any "cost" attributes per data item. Kenyon and Schabanel [7], [22] study the case of a single "cost" attribute per item, assuming again a schedule of infinite size. Liaskos et al. in [9] argued that realistic conditions can require an indefinite number of "cost" attributes per data item (dubbed multi-optimality), and presented a corresponding scheduler. However, [9] reinstated the infinite schedule size assumption, and provided neither proof nor evidence of existence of finite multi-optimal schedules. Furthermore, infinite size implies high complexity [4], meaning that the aforementioned scheduler is fit for infrastructure-based broadcasting only. Both limitations are addressed in the present paper.

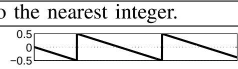
To the best of the authors knowledge, the present study is the first to introduce lightweight, multi-optimal data push for portable devices. The study provides analytical proof and experimental evidence that schedules of minimal size can perform as well as their highly complex counterparts of [9].

III. Analysis

A. System model description

The model of the system is illustrated in Fig. 1 and the notation is summarized in Table I. The system comprises a mobile device that pushes data to a set of wireless neighbors. Each neighbor may have one pending query at a given moment and no caching strategy (discussed as a separate functionality

Table I
SUMMARY OF NOTATION

Symbol	Explanation
Data Items	
$N \in \mathbb{N}$	The total number of data items.
$i = 1 \ldots N$	The data item index.
$p_i \in [0, 1]$	The request probability of item i. $\sum_{i=1}^{N} p_i = 1$.
$l_i \in \mathbb{R}^+$	The size of item i (bytes).
$M \in \mathbb{N}$	The total number of "cost" attributes per item.
$j = 1 \ldots M$	The index of "cost" attributes.
$c_{ij} \in \mathbb{R}^+$	The j^{th} "cost" attribute of the i^{th} data item. Same j implies same type (e.g. copyright cost).
Broadcast (Push) Schedule	
$v_i \in \mathbb{N}$	The number of occurrences of item i in the schedule. Typically derived from the p_i, l_i, c_{ij} inputs.
$r_i \in \mathbb{R}^+$	A ratio expression of v_i as $v_i = [[r_i \cdot \lambda]]$, $\lambda \in \mathbb{R}^+$.
$L \in \mathbb{R}^+$	The total size of the schedule. $L = \sum_{i=1}^{N} v_i \cdot l_i$.
Performance & Operation Conditions	
$\overline{W} \in \mathbb{R}^+$	The mean service time for a user query.
$\overline{C_j} \in \mathbb{R}^+$	The mean (running) cost of the schedule, referring to the j^{th} attribute of the items. Defined as input.
Analysis	
$[[.]]$	Rounding (.) to the nearest integer.
$s_i(\lambda)$	A sawtooth function with period $1/r_i$ 

layer elsewhere [23]). Notice that in push systems client queries are *silent*, i.e. their is no interaction with the push server. A client simply listens to the broadcast stream until successful retrieval of a needed item. Thus, their number is irrelevant.

The server device hosts a database of N data items, $i = 1 \ldots N$, with sizes l_i measured in bytes. The nature of their content is not restricted. They could exemplary be web pages or media files. Each item is associated to a request probability, p_i, expressing the probability of a client query referring to item i. It is clarified that the described item and client models are standard for push systems in general [6], [13], [24], [25].

Additional attributes per item, c_{ij}, $j = 1 \ldots M$, represent any number of other factors, which can be freely set to accommodate any need. For example, c_{i1} may refer to the copyright costs per broadcast of each item, while c_{i2} can be the total battery drain it entails. Related studies have proposed efficient mechanisms for estimating the p_i, c_{ij} attributes. Their values can be calculated adaptively for the specific client set in range [6], or offline from e.g. Facebook-derived statistics. Thus, the present work assumes their values to be known and static.

The device that acts as a push server creates a broadcast schedule. This sequence of data items is broadcasted repeatedly, and each item has v_i periodic occurrences in this sequence. In Fig. 1, for example, item #1 is repeated $v_1 = 3$ times inside the schedule. In general, the v_i values are derived from the item attributes p_i, l_i and the required running "costs", $\overline{C_j}$, i.e. the mean value pertaining to the attributes c_{ij} over all broadcasts in the schedule. Their mathematical expression is as follows:

$$\overline{C_j} = \frac{\sum_{i=1}^{N} v_i \cdot c_{ij}}{\sum_{i=1}^{N} v_i} \tag{1}$$

For example, assume that c_{i1} represents the copyright cost per broadcast of item i. Then, the push server must be able to freely tune the running copyright cost of the schedule, $\overline{C_1}$, by setting it to an acceptable value, e.g. $\mathcal{C}_1$ [7]. Generalizing this concept for each "cost" type $j = 1 \ldots M$, the scheduling server defines the following operation conditions:

$$\overline{C_j} = \mathcal{C}_j,\ \forall j \tag{2}$$

In general, the push server receives the conditions of (2) and the attributes p_i, l_i and c_{ij} as inputs, and produces the periodic broadcast schedule that fulfills the conditions *and* minimizes the mean client waiting time, $\overline{W}$. For any given v_i values, the mean service time per query is given by [1]:

$$\overline{W} = \frac{1}{2} \cdot \left(\sum_{i=1}^{N} v_i \cdot l_i \right) \cdot \left(\sum_{i=1}^{N} \frac{p_i}{v_i} \right) \tag{3}$$

Obtaining the corresponding item occurrences, v_i, is enough for constructing the schedule via the serializer of [4] in $O(L)$ time, L being the total size of the schedule:

$$L = \sum_{i=1}^{N} v_i \cdot l_i \tag{4}$$

Notice that in the $O(.)$ expressions involving L, the l_i attributes are set to 1, in order to yield the integer number of required computational steps.

B. Problem definition

The goal of the present study is to minimize the complexity of the described scheduling process. Particularly, we are interested in obtaining the minimal complexity required to uphold the conditions of (2) while minimizing the mean client waiting time, $\overline{W}$. However, the complexity of the process is proportional to the schedule size, L [4]. Therefore, a sufficient condition is to minimize L. Thus we state the problem definition.

$$\text{Minimize } L(v_i), \text{ for } \{v_i, i = 1 \ldots N\}, \text{ under restrictions:} \tag{5}$$

1) $\overline{C_j(v_i)} = \mathcal{C}_j, j = 1 \ldots M$.
2) $\frac{\partial \overline{W}(v_i)}{\partial v_i} = 0, \{v_i, i = 1 \ldots N\}$.

The first restriction states that the running costs of the schedule should be equal to input values. The second restriction expresses the concurrent minimization of the mean client waiting time.

C. Approach and solution

Our general approach will constitute at producing a v_i solution, assuming a very large schedule size L at first. Then, we will "rewind" the ensuing result via a novel analytical process, making it hold for small schedules as well.

At first, notice that in equations (1) and (3), the item occurrences v_i may be expressed in ratio form, r_i, without any change in the formulas. Thus, assuming a proper multiplicative factor $\lambda > 0$, it holds that:

$$v_i = [[\lambda \cdot r_i]] \tag{6}$$

where [[.]] is the rounding function. The schedule size is then produced from (4) and (6) as:

$$L(\lambda)=\sum_{i=1}^{N} v_i\cdot l_i=\sum_{i=1}^{N}[[\lambda\cdot r_i]]\cdot l_i \tag{7}$$

It is apparent that λ directly regulates the outcome of equation (7), since r_i and l_i are constants. In other words, minimizing L is tantamount to minimizing λ. Thus, for the sake of simplicity, λ will be hereafter referred to as the schedule size.

Assuming schedules of very large size, i.e. $\lambda\to\infty$, equation (6) will hold for any appropriately large λ value:

$$[[\lambda\cdot r_i]] \overset{\lambda\to\infty}{\approx} \lambda\cdot r_i \tag{8}$$

This outcome has constituted the reason for employing the schedule size infinity assumption in the related work. One can treat the item occurrence ratios as continuous variables, using infinitesimal calculus to deduce the case-specific, optimal r_i. Multiplication by any huge number λ and application of rounding produces valid v_i values. However, this trick does not apply to the finite, small schedules we are seeking, i.e. when λ takes small values. To compensate, equation (6) is rewritten as follows:

$$v_i=\lambda\cdot r_i+s_i(\lambda) \tag{9}$$

where $s_i(\lambda)$ is a negated sawtooth function with period $T_i={}^1/_{r_i}$, depicted in Table I. This alternative expression is explained qualitatively as follows. The rounding function [[.]] is a "staircase" function, which can be decomposed as the sum of a ramp function $(\lambda\cdot r_i)$ and a negative sawtooth function $s_i(\lambda)$. The negated sawtooth chips away ramp fragments periodically, producing the original staircase graph. Using the expression of (9) we can deduce expressions of the $\overline{C_j}$ and $\overline{W}$ that hold both for finite and infinite schedules.

Theorem 1. *The mean attribute values, $\overline{C_j}$, can be expressed as functions of the schedule size as:*

$$\overline{C_j(\lambda)}=\overline{C_{j,\infty}}+\sum_{i=1}^{N}\alpha_{ij}\cdot\frac{s_i(\lambda)}{\lambda} \tag{10}$$

where $\overline{C_{j,\infty}}=\frac{\sum_{i=1}^{N} r_i\cdot c_{ij}}{\sum_{i=1}^{N} r_i}$ *and* $\alpha_{ij}=\frac{c_{ij}}{\sum_{i=1}^{N} r_i}-\frac{\sum_{i=1}^{N} r_i\cdot c_{ij}}{\left(\sum_{i=1}^{N} r_i\right)^2}$.

Proof: Let:

$$\widehat{C_j(\lambda)}=\overline{C_j(\lambda)}-\overline{C_{j,\infty}} \tag{11}$$

As $\lambda\to\infty$, $\widehat{C_j(\lambda)}$ is nullified. Using the expressions (1) and (8) it is deduced that:

$$\widehat{C_j(\lambda)}=\frac{\sum_{i=1}^{N} v_i\cdot c_{ij}}{\sum_{i=1}^{N} v_i}-\frac{\sum_{i=1}^{N} r_i\cdot c_{ij}}{\sum_{i=1}^{N} r_i} \tag{12}$$

Using the equivalence of equation (9) and combining the two fractions produces:

$$\widehat{C_j(\lambda)}=\frac{\left(\sum\limits_{i=1}^{N} r_i\right)\cdot\left(\sum\limits_{i=1}^{N} c_{ij}\cdot s_i\right)-\left(\sum\limits_{i=1}^{N} s_i\right)\cdot\left(\sum\limits_{i=1}^{N} r_i\cdot c_{ij}\right)}{\left(\sum_{i=1}^{N} r_i\right)\cdot\left(\sum_{i=1}^{N}\lambda\cdot r_i+\sum_{i=1}^{N} s_i\right)} \tag{13}$$

We notice that eq. (13) is primarily composed of periodic sawtooth functions, s_i. Furthermore, the presence of the linear term $\lambda\cdot r_i$ in the denominator means that $\widehat{C_j(\lambda)}$ fades to zero as λ increases. This observation enables a potential simplification of the equation: it could be expressed as a ratio of a periodic function divided by a continuous, fading component. We will first express the simplification technique generally, and then proceed to apply it to equation (13).

Simplification Technique: Assume a pseudoperiodic fading function $f(\lambda)$, for which it holds that $\lim\limits_{\lambda\to\infty} f(\lambda)=0$. Let $\Pi(\lambda)$ be a multiplicative factor such that $\lim\limits_{\lambda\to\infty} f(\lambda)\cdot\Pi(\lambda)\neq 0,\infty$. Then, $f(\lambda)$ can be approximated as $f(\lambda)={}^{\lim_{\lambda\to\infty}(f(\lambda)\cdot\Pi(\lambda))}/_{\Pi(\lambda)}$. The equivalence may also be precise, depending on the chosen multiplicative factor.

Application: Equation (13) fulfills the criteria of the technique, since a multiplicative factor of the form $\Pi(\lambda)=\lambda$ produces:

$$\lim_{\lambda\to\infty}\widehat{C_j(\lambda)}\cdot\lambda=\frac{\sum\limits_{i=1}^{N} r_i\left(\sum\limits_{i=1}^{N} c_{ij}\cdot s_i\right)-\sum\limits_{i=1}^{N} s_i\left(\sum\limits_{i=1}^{N} r_i\cdot c_{ij}\right)}{\left(\sum_{i=1}^{N} r_i\right)^2}, \tag{14}$$

which is not zero or infinite in the general case. Therefore, $\widehat{C_j(\lambda)}$, can be expressed as:

$$\widehat{C_j(\lambda)}=\frac{\lim\limits_{\lambda\to\infty}\widehat{C_j(\lambda)}\cdot\lambda}{\lambda}=\sum_{i=1}^{N}\left(\frac{c_{ij}}{\sum\limits_{i=1}^{N} r_i}-\frac{\sum\limits_{i=1}^{N} r_i\cdot c_{ij}}{\left(\sum\limits_{i=1}^{N} r_i\right)^2}\right)\cdot\frac{s_i(\lambda)}{\lambda} \tag{15}$$

leading to equation (10), QED. ■

An interesting observation regarding equation (10) is that the term $\sum_{i=1}^{N}\alpha_{ij}\cdot\frac{s_i(\lambda)}{\lambda}$ can take both positive and negative values with regard to λ (since $-0.5<s_i<0.5$). Therefore, it can also be nullified for one or more, finite λ values. At this point, the resulting running "costs", $\overline{C_j(\lambda)}$, coincide with the performance of infinite schedules, $\overline{C_{j,\infty}}$. In other words, a finite, low complexity schedule can fulfill any operational restrictions regarding its running costs, as good as a sizeable, highly complex one. Choosing the smallest λ that nullifies the term $\sum_{i=1}^{N}\alpha_{ij}\cdot\frac{s_i(\lambda)}{\lambda}$ yields the smallest possible schedule size and, therefore, the smallest possible scheduling complexity [4].

However, according to the problem definition (5), the optimal schedule must also yield minimal mean client waiting times. Therefore, it is essential to derive a similar theorem for equation (3), and ensure that the new, optimal λ-values coincide with those of Theorem 1.

Theorem 2. *The mean service time, $\overline{W(\lambda)}$, can be expressed as a function of the schedule size as:*

$$\overline{W(\lambda)}=\overline{W_\infty}+\sum_{i=1}^{N} b_i\cdot\frac{s_i(\lambda)}{\lambda} \tag{16}$$

where $\overline{W_\infty}=\frac{1}{2}\cdot\left(\sum_{i=1}^{N} r_i\cdot l_i\right)\cdot\left(\sum_{i=1}^{N}\frac{p_i}{r_i}\right)$ *and* $b_i=l_i\cdot\left(\sum_{i=1}^{N}\frac{p_i}{l_i}\right)-\frac{p_i}{r_i^2}\cdot\left(\sum_{i=1}^{N} r_i\cdot l_i\right)$.

Proof: Similarly to Theorem 1, the quantity $\widehat{W(\lambda)}$ is defined as:

$$\widehat{W(\lambda)} = \overline{W(\lambda)} - \overline{W_\infty} \tag{17}$$

in order to comply with the $\lim_{\lambda\to\infty} \hat{W(\lambda)} = 0$ condition of the previously described simplification technique. Thus:

$$\widehat{W(\lambda)} = \frac{\sum_{i=1}^{N} v_i \cdot l_i}{2}\left(\sum_{i=1}^{N}\frac{p_i}{v_i}\right) - \frac{\sum_{i=1}^{N} r_i \cdot l_i}{2}\left(\sum_{i=1}^{N}\frac{p_i}{r_i}\right) \tag{18}$$

To facilitate the application of the technique, equation (18) is rewritten by expanding the $\sum(.)$ products as:

$$\widehat{W(\lambda)} = \frac{1}{2}\cdot\sum_{i=1}^{N}\sum_{k=1}^{N} w_{ik},\ w_{ik} = p_i \cdot l_k \cdot \left(\frac{v_k}{v_i} - \frac{r_k}{r_i}\right) \tag{19}$$

We examine the quantity:

$$w_{ik} + w_{ki} = p_i \cdot l_k \cdot \left(\frac{v_k}{v_i} - \frac{r_k}{r_i}\right) + p_k \cdot l_i \cdot \left(\frac{v_i}{v_k} - \frac{r_i}{r_k}\right) \tag{20}$$

Combining the two quantities into one fraction produces through (9):

$$\lim_{\lambda\to\infty}(w_{ik} + w_{ki})\cdot\lambda = \frac{\left(l_i p_k r_i^2 - l_k p_i r_j^2\right)(r_k s_i - r_i s_k)}{r_i^2 r_k^2} \tag{21}$$

We apply the simplification technique of Theorem 1 on $(w_{ik} + w_{ki})$, with $\Pi(\lambda) = \lambda$. We produce:

$$w_{ik} + w_{ki} = \frac{1}{\lambda}\cdot\frac{\left(l_i \cdot p_k \cdot r_i^2 - l_k \cdot p_i \cdot r_k^2\right)\cdot(r_k \cdot s_i - r_i \cdot s_k)}{r_i^2 \cdot r_k^2} \tag{22}$$

Finally, from (19) we deduce:

$$\widehat{W(\lambda)} = \sum_{i=1}^{N}\left[l_i \cdot \left(\sum_{i=1}^{N}\frac{p_i}{l_i}\right) - \frac{p_i}{r_i^2}\cdot\left(\sum_{i=1}^{N} r_i \cdot l_i\right)\right]\frac{s_i}{\lambda} \tag{23}$$

which is inline with the statement of Theorem 2, QED. ■

Theorem 2 also states that the performance of an infinite schedule can be obtained by finite ones, since the term $\sum_{i=1}^{N} b_i \cdot \frac{s_i(\lambda)}{\lambda}$ has varying sign. An interesting remark is that, since $\widehat{W(\lambda)}$ can be equal to zero for some finite λ (i.e. finite schedule), the overall performance trade-off can also be nullified. However, a major point of [4], was that finite schedules are generally expected to be suboptimal, i.e. yield larger mean waiting times for the clients than their infinite counterparts. The addition of extra optimization objectives actually relaxes this restriction. Each objective adds a performance tuning point to the overall operation of the system. Essentially, the additional optimization objectives share the effects of the performance trade-off. At the same time, the divergence per optimization objective diminishes as the number of objectives increases. This remark is also expressed formally in the following sub-section.

D. Exceptions

The preceding analysis operated under the assumption that equations (14) and (21) are not nullified, as this would violate the necessary conditions of the employed simplification technique. However, there exist three distinct cases which require special handling.

Firstly, (14) and (21) can be nullified when $l_i, c_{ij}, r_i \to 0, \forall i, \forall j$. In this extreme case where *all* items have small attributes, we can simply normalize their values, e.g. in $[0, 1]$. Thus, the problem is resolved without loss of generality or precision.

Secondly, equation (14) is nullified for all i and λ, $\forall j$ when it holds:

$$\sum_{i=1}^{N} r_i\left(\sum_{i=1}^{N} c_{ij}\cdot s_i(\lambda)\right) - \sum_{i=1}^{N} s_i(\lambda)\left(\sum_{i=1}^{N} r_i \cdot c_{ij}\right) = 0 \Rightarrow$$
$$\sum_{i=1}^{N} s_i(\lambda)\left[\left(\sum_{i=1}^{N} r_i\right)\cdot c_{ij} - \left(\sum_{i=1}^{N} r_i \cdot c_{ij}\right)\right] = 0 \Rightarrow$$
$$\left(\sum_{i=1}^{N} r_i \cdot c_{ij}\right) = \left(\sum_{i=1}^{N} r_i\right)\cdot c_{ij},\ \forall i, \forall j \tag{24}$$

In other words, equation (14) is nullified when the attributes c_{ij} are constant over all i, $\forall j$. However, in this case equation (1) states that the mean costs $\overline{C_j}$ would be fixed, with no tunability. Besides, assigning constant c_{ij} attributes to the data items is equivalent to disregarding these optimization criteria altogether.

Thirdly, equation (21) is nullified when:

$$l_i p_k r_i^2 = l_k p_i r_k^2 \Rightarrow \frac{r_i}{r_k} = \frac{\sqrt{\frac{p_i}{l_i}}}{\sqrt{\frac{p_k}{l_k}}} \Rightarrow r_i \propto \sqrt{\frac{p_i}{l_i}},\ \forall i \tag{25}$$

Equation (25) is the expression of the *"golden" condition,* which defines the item occurrence ratio, r_i, that minimizes the mean service time with no regard to other criteria [1]. In other words, eq. (21) is nullified when we target exclusively the minimization of the mean service time (single objective-optimal schedule). This outcome is once again contradicting to the initial hypothesis of multi-obective optimization. Thus, it is deduced that:

Corollary 1. *A single objective-optimal finite schedule cannot offer the exact performance of the corresponding infinite one.*

On the other hand, multi-optimal schedules have potential for perfect performance. We proceed to request the synchronization of the optimal schedule sizes derived from Theorems 1 and 2. For a valid, finite schedule size λ^*, it must hold that:

$$\begin{cases} \overline{W(\lambda^*)} = \overline{W_\infty} \\ \overline{C_j(\lambda^*)} = \overline{C_{j,\infty}},\ \forall j \end{cases} \tag{26}$$

Therefore, from equations (10) and (16) we deduce the following condition.

Corollary 2. *A finite multi-optimal schedule of size λ^* offers exactly the same performance as the corresponding infinite schedule if it holds:*

$$\mathcal{S}\cdot\mathcal{A} = \mathbb{O} \tag{27}$$

where $\mathcal{S} = \begin{bmatrix} s_1(\lambda^*) & s_2(\lambda^*) & \cdots & s_N(\lambda^*) \end{bmatrix}$,

$$\mathcal{A} = \begin{bmatrix} b_1 & \alpha_{11} & \cdots & \alpha_{1M} \\ b_2 & \alpha_{21} & \cdots & \alpha_{2M} \\ b_3 & \alpha_{31} & \cdots & \alpha_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ b_N & \alpha_{N1} & \cdots & \alpha_{NM} \end{bmatrix}$$ *and $\mathbb{O}$ is the null matrix.*

Equation (27) represents a $N \times (M+1)$ indefinite system in the general case. A practical approach for solving it is the following. One can solve only equation $\sum_{i=1}^{N} b_i \cdot s_i(\lambda^*) = 0$, producing a set of valid λ^* values sorted by ascending order. The smallest value that fulfills the remaining conditions, $\sum_{i=1}^{N} \alpha_{ij} \cdot s_i(\lambda^*) = 0$, is chosen as the solution to the system. Approximate solutions to equation (27) can also be used. For example, λ^* values that yield $\left|\sum_{i=1}^{N} \alpha_{ij} \cdot s_i(\lambda^*)\right| < \epsilon$ and $\left|\sum_{i=1}^{N} b_i \cdot s_i(\lambda^*)\right| < \epsilon$ can be considered valid, where ϵ is a user-defined acceptable error. Numerical equation solvers, for example, typically operate in this manner [26].

E. Complexity considerations

As described in Section III, the data push process comprises two parts. Firstly, the optimal number of periodic item occurrences v_i, $i = 1 \ldots N$ are defined. Then, a serializer produces the final broadcast sequence according to these v_i values.

Traditionally, the serialization is the bottleneck of the whole process [24], requiring $O(L)$ complexity [4]. Prior to the present study, periodic serialization required a schedule of "infinite" size L [24], i.e. excessively large schedule sizes and complexity. The importance of Corollary 2 is that it keeps L minimal. In practice, L is typically kept at $O(N)$ levels. (E.g. $2 \cdot N \to 5 \cdot N$ operations in Section IV).

The complexity of the calculations for the optimal v_i, $i = 1 \ldots N$ values is that for constructing the array $\mathcal{A}$ of Corollary 2 and solving the equation $\sum_{i=1}^{N} b_i \cdot s_i(\lambda^*) = 0$ for $\lambda^* > 0$. The $b_i, i = 1 \ldots N$ elements of $\mathcal{A}$ can be calculated in a total of $O(N)$ steps (Theorem 2). The α_{ij} elements can be calculated in $O(M \cdot N^2)$ operations in total (Theorem 1). Finally, if a numerical solver requires B steps to converge with precision ϵ, solving $\sum_{i=1}^{N} b_i \cdot s_i(\lambda^*) = 0$ for λ^* takes $O(N \cdot B)$ operations. Thus, the $O(M \cdot N^2)$ factor defines the complexity of the whole process.

However, we target push servers that are implemented on portable devices, such as smartphones. The wireless connectivity range of such devices is $\backsim 100m$, and thus the pushed data are expected to have a high degree of local significance. Thus, pushing $N = 1000$ data items with $M = 10$ attributes should be more than enough for most application scenarios. The required $\backsim 10^7$ operations are well within the capabilities of most modern smartphones, equipped with up to four CPUs running at $\backsim 1GHz$.

Another major consideration for services running on portable devices is the induced power consumption. The proposed algorithm is tunable in this aspect, as will be demonstrated in Section IV. In a straightforward manner, we add a "cost" attribute per data item that expresses the energy required for a single broadcast. The energy drain rate can then be set to a value fit for the hosting device, by introducing an "average cost" restriction in the condition set 1 of the objective (5).

IV. SIMULATIONS

In this section we present a realistic application scenario of mobile, push-based broadcasting, implemented in $MATLAB^{TM}$ [27]. The goals of the ensuing simulations are:

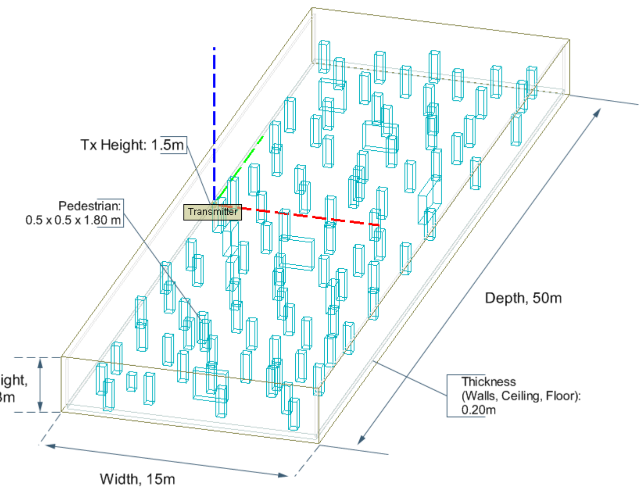


Figure 2. Topology and geometry of the simulated system. A transmitter/broadcaster is placed in the middle of a $50m$-long corridor. A number of pedestrians traverse the corridor and listen to the broadcast, moving in directions parallel to the side walls.

- To check the validity of the preceding analysis in terms of equivalence between finite and infinite schedules.
- To demonstrate the efficiency of the proposed, finite schedules in a wireless, mobile environment by studying aspects such as energy-efficiency, service ratio and mean service time.
- To demonstrate the realizability gains of finite schedules in mobile devices, with regard to the original, infinite schedules.

A. Application scenario

The scenario assumes one broadcast server hosted by a mobile device. The server continuously transmits a periodic schedule of a set of data items. The receivers are passers-by, interested in the broadcasted data. The server and the clients are placed in a rectangular corridor, as depicted in Fig. 2. The server is placed at the middle and side of the corridor. A number of approximately 120 pedestrians are present in the examined part of the corridor at any given moment. They walk in directions parallel to the walls, either upwards or downwards. Their absolute walking speeds, s_w, are picked at random, by the following probability distribution:

$$P(s_w) = \frac{1}{2.5} \cdot e^{s_w \cdot 1/2.5} \, (m/sec) \quad (28)$$

limited in $[0.5,\, 5]m/sec$, simulating a walking crowd with a few running pedestrians, while collisions are ignored. At any given moment while traversing the corridor, a client begins to receive a needed item from the broadcast stream. Since all data items are broadcasted periodically, the waiting time prior to reception start is in the range $[0,\, T_{B_i}]$, T_{B_i} being the broadcast period of item i. The reception of the needed item by a given client may be successful or not, depending on the signal quality. If a client reaches the end of the corridor while still reading a data item from the broadcast stream, we assume that he waits for the item transmission to finish, and then exits the corridor. When the client exits the corridor, a new one is generated at the opposite side, keeping the number of clients approximately constant at all times. In case of reception failure, queries are not re-posed. At the client side, the metrics of interest are the query service ratio and the mean service time of successfully answered queries. Each simulation run lasts for a total of $3,000$ queries (successful or not) and is repeated 100 times, typically achieving better than 90% confidence for the logged metrics.

The server hosts a database of $N = 20$ items with sizes randomly picked from the range $[1,\,10]Mbit$, fitting in the application storage space of any modern smartphone. Since the client set is being constantly replenished (due to mobility), these items are reusable, justifying the choice of periodic broadcasting. The request probability of the items follows the Zipf popularity distribution, a popular choice in related studies (e.g. [1], [9], [18], [21]). According to it:

$$p_i = \frac{i^{-\theta}}{\sum_{i=1}^{N} i^{-\theta}} \tag{29}$$

where $\theta > 0$ is the skewness regulation parameter, which is typically set to $\theta = 0.95$ for web data. The server uses power control and transmits each item with a different power level, POW_i. The goal is to broadcast bigger items with greater power, ensuring a high transmission success ratio. The POW_i values were produced deterministically from a simple formula, as $POW_i(dB_m) = \frac{20}{9} \cdot (l_i(Mb) - 1)$[1] and are assumed to reflect the total wireless power cost of the transmission. Finally, it is assumed that each data item has a copyright cost, and thus the server pays a certain monetary amount of MON_i per broadcast of item i. The units of MON_i are arbitrary and the values are random and normalized in $[0,\,1]$. In this manner, the setup examines both dependent (POW_i vs l_i) and independent (MON_i vs any other) item attributes.

Being a mobile device, the server seeks to control the energy expenditure rate, i.e. the mean of POW_i over all broadcasts. Furthermore, the mean monetary expense rate (mean of MON_i over all broadcasts) must also be in check. The POW_i and MON_i correspond to the c_{ij} attributes of Corollary 2, for $j = 1, 2$. Finally, the corresponding schedule must require as little resources and time for construction as possible, a factor directly proportional to its aggregate size [4]. Thus, the server operates as follows.

A fitting energy expenditure rate and an acceptable monetary cost are selected. Then, equations (2) and (1) are passed to the numerical solver of $MATLAB$ (fsolve)[2]. Thus, the solver produces the instructions for the creation of an infinite corresponding schedule. These instructions come in the form of item occurrences, $v_i^{(inf)}$. Through Corollary 2, the server then employs the present study to pinpoint finite schedule sizes (λ values) with equal performance. Through equation (6) these λ values are transformed to actual item occurrences. Finally, the serializer of [4] receives these item occurrences and item sizes as inputs and produces the final broadcast schedule.

Finally, we note that it is possible to change the p_i, POW_i and MON_i attributes in real-time. However, this would imply posting queries to a server or gathering data in an accelerated rate. While not prohibited, this course of action could lead to redundant energy consumption. Targeting lightweight implementation, we assume that p_i are derived statically, once, from e.g. Facebook data, while POW_i are calculated directly from the item sizes. Likewise, MON_i are assumed to change slowly over time and need not be updated too often.

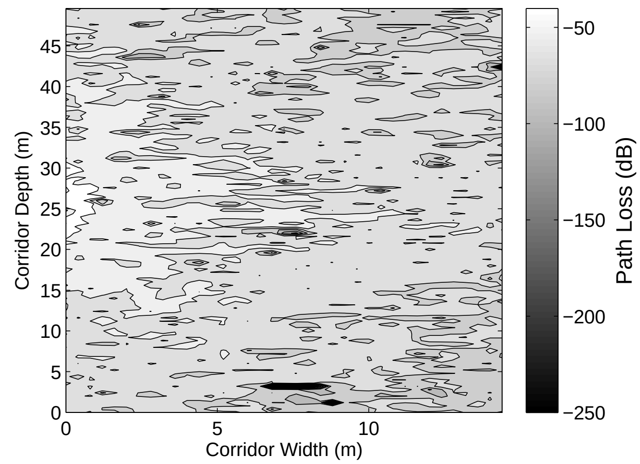


Figure 3. Snapshot of the path loss at $1.5m$ from the floor of the corridor (ray traced). The presence of the pedestrians and the wall echos result into a non-trivial coverage map. The mean path loss is roughly $-75dBm$ across the map.

B. Channel Configuration

The simulation uses full 3D ray tracing to calculate the power delay profile at a given receiver [28]. A number of 8 ray interactions are allowed, while diffractions are ignored. The electromagnetic properties of the used materials are given in Table II. The presence of pedestrians and the wall echos result into non-trivial path loss maps, as presented in Fig. 3. In addition, the study assumes the typical parameters of a single 802.11 channel given in Table III.

Table II
ELECTROMAGNETIC PROPERTIES OF MATERIALS USED.

Description	Permittivity (F/m)	Conductivity ($\Omega \cdot m$)
Walls (Dry wall)	2.8	0.001
Floor, Ceiling (Concrete)	15.0	0.015
Human Body	81.0	20.0

Fading is handled deterministically through the ray tracing. Rays that fail to reach the receiver within the guard interval are considered as interference when calculating the time-variant

[1] Given that $l_i \in [1, 10]$, the formula was crafted to correspond to 0 to $20dBm$ Tx power range. The typical path loss for the setup of Fig. 2 is $-75dBm$ (see Fig. 3). Thus, the Rx power level ranges roughly from $-75dBm$ to $-55dBm$, which is within the assumed sensitivity of the receiver ($-100dBm$, Table III).

[2] Notice that the solution must comply with the conditions: i) $v_i > 0$, ii) $\partial\overline{W}/\partial v_i = 0$, iii) $L > L_\infty$. The L_∞ quantity is a very large, arbitrarily set threshold. The heuristic of [9] could also be used as an alternative to the MATLAB *fsolve* numerical solver.

Table III
CHANNEL CONFIGURATION PARAMETERS

Parameter	Setting
Antennas (Tx, Rx)	Half-wave dipoles
Carrier Frequency	$2.4GHz$
Modulation	$QAM-16$
Rx Sensitivity	$-100dBm$
Channel Bandwidth	$1MHz$
Symbol Duration	$3.2\mu sec$
Guard Interval	$0.8\mu sec$
FEC gain	$3dB$
Noise Temperature	300^oK

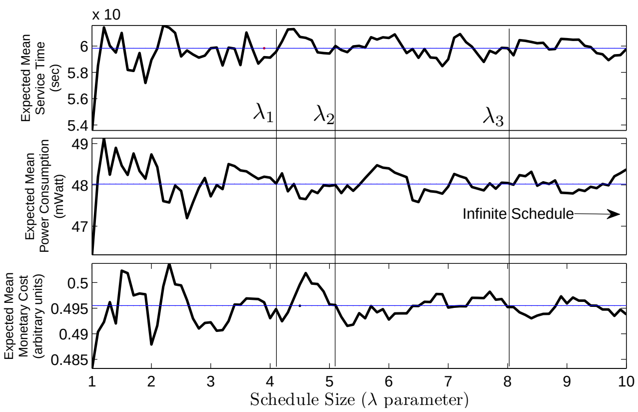


Figure 4. Overview of the process for pinpointing the optimal, finite schedules. The graphs correspond to a request for a schedule that yields a mean power consumption (wireless module) of $48mW$ and a mean monetary cost of 0.4951. The horizontal lines correspond to the performance of an infinite schedule. Theorem 2 produces the uppermost graph of the mean service time versus the schedule size. Theorem 1 yields the second and third graphs for the mean energy consumption and monetary cost. The optimal, finite schedules are defined as the concurrent intersection points of the horizontal lines and the three graphs. An intersecting tolerance of $\epsilon = 1\%$ is used.

signal-to-interference-plus-noise ratio ($SINR$). The calculated $SINR$ at a receiver is transformed to E_b/N_0 format as:

$$E_b/N_0 = SINR \cdot \frac{Bandwidth}{Net\,Bitrate} \tag{30}$$

The net bitrate (i.e. the effective bitrate) can be easily derived from the symbol duration and guard interval as $^{4\,bits\,per\,symbol}/_{3.2+0.8\mu sec} = 1Mbps$. Finally, the Bit Error Rate is calculated as:

$$BER = \frac{3}{8} erfc(\sqrt{\frac{2}{5} \cdot E_b/N_0}) \tag{31}$$

for Gray coded, $QAM-16$ (i.e. $k=4$) with additive white Gaussian noise [29, p.193]. $Erfc(.)$ is the complementary error function. The BER then defines the success of the symbol's (and the containing item's) reception. Conceptually, a random number is picked uniformly in $[0,1]$ for each received bit. If it is less than the BER, the bit is received erroneously, and so is the corresponding data item at the receiving device.

C. Simulation Runs and Results

We examine the performance of the proposed finite schedules and the corresponding infinite schedule for multiple

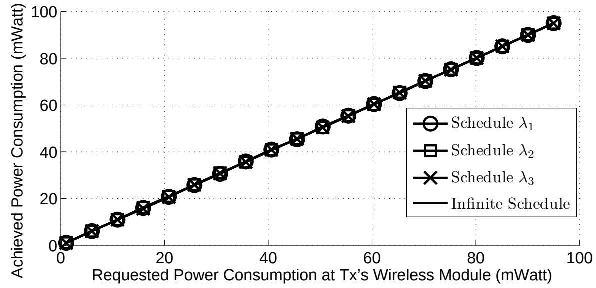


Figure 5. Comparison of requested and achieved mean power consumption at the transmitter's wireless module. All three finite schedules achieve the same performance as the infinite schedule.

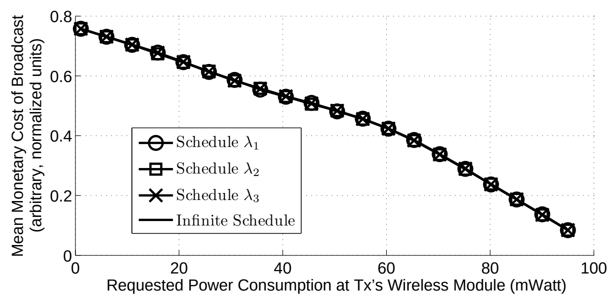


Figure 6. Comparison of mean monetary costs pertaining to the finite and infinite schedules. The x-axis corresponds to those of Fig. 5, 7 and 8. In essence, the finite schedules offer exactly the same performance as the infinite schedule, in every examined case.

requested energy expenditure (running) rates at the transmitter's wireless module. These requests range from $1mW$ to $100mW$ and form the x-axis of the following graphs. Each energy expenditure request is accompanied by a corresponding monetary expense rate, presented and discussed separately in Fig. 6. For each combination of these two quantities, the standard, MATLAB numerical solver produces the infinite schedule construction instructions, $v_i^{(inf)}$, as discussed in Section IV-A. These instructions also correspond to the minimal mean service time for the given requests.

Fig. 4 visualizes the effect of Corollary 2 in the exemplary case of $48mW$ as the energy expenditure rate. Equations (10) and (16) are plotted versus the λ parameter. It is apparent that certain schedule sizes offer theoretically the same performance as the corresponding infinite schedule. We choose the first three, fittest λ values and sort them by ascending order, λ_1, λ_2, λ_3. Their fitness is based on the total absolute divergence from the horizontal lines. The remaining figures compare the performance of these finite schedules to that of the infinite one.

Firstly, all three finite schedules achieve the nominal requested energy expenditure, as presented in Fig. 5. Each achieved rate is calculated over all broadcasts in the simulation scenario. Thus, the finite schedules achieve the same performance as the infinite one from the aspect of compliance with energy expenditure requests. The same conclusion applies to the achieved monetary expenditure rate presented in Fig. 6.

Concerning the client-side metrics of mean service time and service ratio, Fig. 7 shows that the finite schedules achieve optimal performance in any of the examined cases. It is noted that the convex form of the plots is a result of the energy expenditure requests. When the server requests very low energy expenditure rates, the schedule tends to contain more occurrences of the items with the lowest POW_i attribute. Thus, less "air-time" is left for the remaining items, causing the increase in the service time. The same principle applies to requests for high rates.

The service ratio is not affected by the use of finite schedules (Fig. 8). The finite schedules do not alter the POW_i values per item i, and therefore do not make the successful transmission of an item more or less probable. Thus, the finite and the infinite schedules achieve the same service ratio ($\approx 95\%$).

Most importantly, the finite schedules achieve the same performance as the infinite one with orders of magnitude smaller schedule sizes, while achieving optimal performance from all examined aspects (Fig. 9). In Fig. 9b the λ-ranking is also shown to follow the real size ranking (equation (7)), justifying the use of λ as the size-regulating variable. The dramatic decrease in the required schedule size enables mobile devices to act as local, push-based broadcast servers. The original sizes of $\sim 10^{11} GBytes$ would require prohibitively high calculation times and energy to be constructed, while their caching would be impossible. Segmental construction and use would be the only option. The aggregate energy required to perform the scheduling would be prohibitively high nonetheless.

On the other hand, the presented analysis led to a decrease in the total schedule size by a factor of 10^{10}, with negligible or no impact on the performance. Notice that the conditions for optimal performance can be relaxed further. For example, the intersection tolerance in Fig. 4 could be set to 5% or 10% limiting the final schedule size even further.

The use of the energy and monetary expenditure attributes was meant as an appealing application example. However, Corollary 2 enables the use of an indefinite number of attributes (c_{ij}) per item. Thus, it becomes apparent that the presented analysis can be used for incorporating complete hardware, software, or economic models of mobile devices in the broadcast scheduling process.

V. Conclusion

The present study enabled the use of push-based data dissemination in networks of mobile devices. Data push was traditionally considered as an infrastructure approach, due to its heavy computational overhead and inflexibility in accounting for anything but basic, service time concerns. Through mathematical analysis, the study reduced the complexity of data push by orders of magnitude, while allowing for an unlimited number of optimization objectives at the same time. Realistic simulations demonstrated the benefits of the analysis in practice: The proposed approach allowed for a mobile device to balance energy-efficiency, small client service times and copyright costs as efficiently as any infrastructure-based solution.

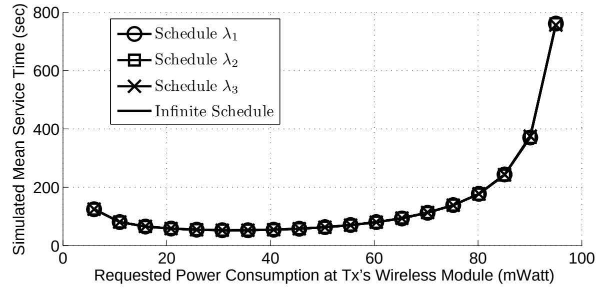


Figure 7. Mean query service times, achieved by the finite and infinite schedules. The x-values correspond to those of Fig. 5. The finite schedules achieve exactly the same performance as the infinite one. Notice that when the requested mean power consumption is very small or very large, the schedule reduces to repeated broadcasts of the items corresponding to the smallest and largest transmission powers respectively. Thus, the service time for other requests is increasing, resulting into a higher mean service time. This fact explains the convex form of the graph.

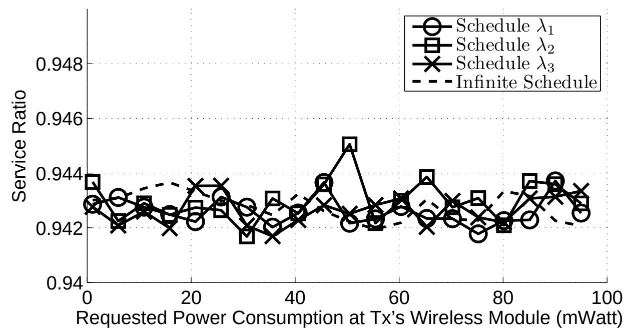


Figure 8. Mean service ratios achieved by the finite and infinite schedules. The finite schedules are equivalent to the infinite one in this aspect as well.

References

[1] N. H. Vaidya and S. Hameed, "Scheduling data broadcast in asymmetric communication environments," *Wireless Networks*, vol. 5, no. 3, pp. 171–182, 1999.

[2] S. Muthukrishnan, *Data streams: Algorithms and applications*, ser. Foundations and trends in theoretical computer science. Boston and MA: Now Publishers, 2005, vol. v. 1, no. 2.

[3] C. Liaskos, S. Petridou, G. Papadimitriou, P. Nicopolitidis, and A. Pomportsis, "On the Analytical Performance Optimization of Wireless Data Broadcasting," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 2, pp. 884–895, 2010.

[4] C. Liaskos, S. Petridou, and G. Papadimitriou, "Towards Realizable, Low-Cost Broadcast Systems for Dynamic Environments," *IEEE/ACM Transactions on Networking*, vol. 19, no. 2, pp. 383–392, 2011.

[5] D. Serpanos, "Scheduling objects in broadcast systems with energy-limited clients," *Computer Communications*, vol. 27, no. 10, pp. 1036–1042, 2004.

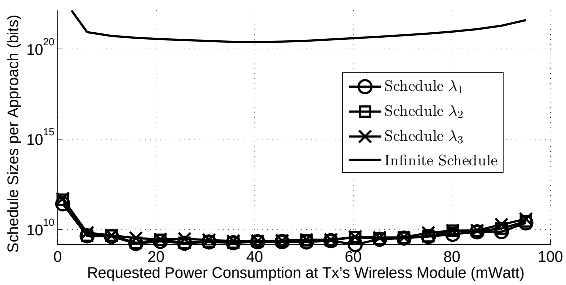
[6] V. Kakali, G. Papadimitriou, P. Nicopolitidis, and A. Pomportsis, "A New Class of Wireless Push Systems," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4529–4539, 2009.

[7] C. Kenyon and N. Schabanel, "The Data Broadcast Problem with Non-Uniform Transmission Times," *Algorithmica*, vol. 35, no. 2, pp. 146–175, 2002.

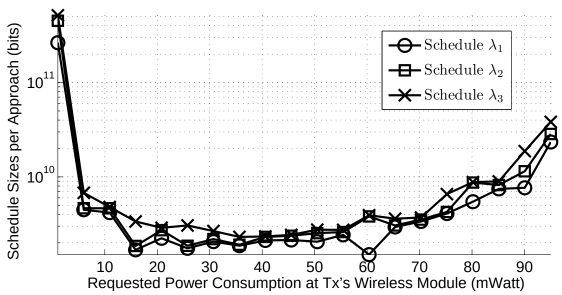
[8] S. Jiang and N. H. Vaidya, "Scheduling data broadcast to "impatient" users," in *Proceedings of the 1st ACM International Workshop on Data Engineering for Wireless and Mobile Access (MoBiDe'99), Seattle, Washington, United States*. ACM, 1999, pp. 52–59.

[9] C. K. Liaskos, A. Xeros, G. I. Papadimitriou, M. Lestas, and A. Pitsillides, "Broadcast Scheduling with multiple concurrent costs," *IEEE Transactions on Broadcasting*, vol. 58, no. 2, pp. 178–186, 2012.

[10] D. Aksoy and M. Franklin, "R$\times$W: a scheduling approach for large-scale on-demand data broadcast," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 846–860, 1999.



(a) While offering no advantage in terms of performance (see Fig. 5-6), the infinite schedule is several orders of magnitude greater than any of the finite ones. Such a schedule could not be constructed in time, or cached by a mobile device. On the other hand, the finite schedules proposed by the present paper constitute viable choices. Notice that the proposed, finite schedules offer the same performance as the infinite one. By using higher intersection tolerance in Fig. 4, one can produce even smaller schedules.



(b) Detail of Fig. (9a) pertaining to the ranking of the finite schedules. Notice that the sizes (y-axis) are calculated through equation (7). However, their ranking obeys to their λ-ordering in all cases ($\lambda_1 < \lambda_2 < \lambda_3$).

Figure 9. Comparison of schedule sizes pertaining to each of the compared solutions.

[11] R. Unni and R. Harmon, "Perceived effectiveness of push vs. pull mobile location-based advertising," *Journal of Interactive Advertising*, vol. 7, no. 2, pp. 28–40, 2007.

[12] A. Bar-Noy, "Optimal Broadcasting of Two Files over an Asymmetric Channel," *Journal of Parallel and Distributed Computing*, vol. 60, no. 4, pp. 474–493, 2000.

[13] J. Gecsei, *The Architecture of Videotex Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1983.

[14] C.-J. Su, L. Tassiulas, and V. J. Tsotras, "Broadcast scheduling for information distribution," *Wireless Networks Journal*, vol. 5, no. 2, pp. 137–147, 1999.

[15] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast disks," *ACM SIGMOD Record*, vol. 24, no. 2, pp. 199–210, 1995.

[16] B. Zheng, X. Wu, X. Jin, and D. L. Lee, "TOSA: a near-optimal scheduling algorithm for multi-channel data broadcast," in *Proceedings of the 6th International Conf. on Mobile Data Management (MDM'05), Ayia Napa, Cyprus, May*, 2005, pp. 29–37.

[17] F. J. Ovalle-Martinez, J. S. González, and I. Stojmenović, "A parallel hill climbing algorithm for pushing dependent data in clients-providers-servers systems," *Mob. Netw. Appl*, vol. 9, pp. 257–264, 2004. [Online]. Available: http://dx.doi.org/10.1145/1012215.1012217

[18] S. Kim and S. H. Kang, "Scheduling Data Broadcast: An Efficient Cut-Off Point Between Periodic and On-Demand Data," *IEEE Communications Letters*, vol. 14, no. 12, pp. 1176–1178, 2010.

[19] K. Mouratidis, S. Bakiras, and D. Papadias, "Continuous Monitoring of Spatial Queries in Wireless Broadcast Environments," *IEEE Transactions on Mobile Computing*, vol. 8, no. 10, pp. 1297–1311, 2009.

[20] H.-Y. Shin, S.-M. Wu, and J.-Y. Chen, "Energy-efficient algorithm to improve the performance of wireless data broadcasting," in *Proceedings of the 2011 IEEE International Conference on Machine Learning and Cybernetics (ICMLC'11), Guilin, China, September*, 2011, pp. 1226–1231.

[21] H.-Y. Shin, "Exploiting skewed access and energy-efficient algorithm to improve the performance of wireless data broadcasting," *Computer Networks*, vol. 56, no. 4, pp. 1167–1182, 2012.

[22] N. Schabanel, "The Data Broadcast Problem with Preemption," in *Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS'00), Lille, France, January*, H. Reichel and S. Tison, Eds. Springer, 2000, pp. 181–192.

[23] J. Xu, D.-L. Lee, Q. Hu, and W.-C. Lee, "Data Broadcast," in *Wiley Series on Parallel and Distributed Computing*, I. Stojmenović and A. Y. Zomaya, Eds. New York and USA: John Wiley & Sons, Inc, 2002, pp. 243–265.

[24] S. Hameed and N. H. Vaidya, "Efficient algorithms for scheduling data broadcast," *Wireless Networks*, vol. 5, no. 3, pp. 183–193, 1999.

[25] C. K. Liaskos and G. I. Papadimitriou, "Optimal Periodic Scheduling Under Multimodel Per-Item Constraints in Wireless Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1071–1080, 2012.

[26] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *SIAM Journal on Applied Mathematics*, vol. 11, no. 2, p. 431, 1963.

[27] MathWorks-Inc, "MATLAB version (R2010a)," 2010. [Online]. Available: http://www.mathworks.com/products/matlab/

[28] Radioplan, "Radio Wave Propagation Simulator SE 5.2," 2011. [Online]. Available: http://www.actix.com/our-products/radioplan/index.html

[29] J. R. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital communication*, 3rd ed. Boston: Kluwer Academic Publishers, 2004.

Christos Liaskos received the Diploma in Electrical and Computer Engineering from the Aristotle University of Thessaloniki (AUTH), Greece in 2004, the MSc degree in Medical Informatics in 2008 from the Medical School, AUTH and the Ph.D. degree in Computer Networking from the Dept. of Informatics, AUTH in 2014. He is currently a postdoctoral research fellow at the Foundation Of Research and Technology, Hellas (FORTH).

Ageliki Tsioliaridou received the Diploma (2004) and the Ph.D degree (2010) in Electrical and Computer Engineering from the Democritus University of Thrace (DUTH), Greece. Her research in Computer Networks focuses on congestion control, fair allocation of network resources, as well as convergence potential and speed of routing protocols. She has contributed to a number of EU, ESA and National research projects. She is currently a postdoctoral fellow at Foundation Of Research and Technology, Hellas (FORTH).

Georgios Papadimitriou (Senior Member, IEEE) is a Full Professor in the Department of Informatics, Aristotle University, Greece. He is the Director of the Computer Architecture and Communications Laboratory. He holds a PhD degree (1994) and a 5-year Diploma degree (1989) in Computer Engineering and Informatics from the University of Patras. His major research interests are: Wireless Networks, Optical Networks, Learning Algorithms and Application of Learning Algorithms in Communication Networks. He has published 107 papers in peer-reviewed journals (42 in IEEE Journals) and 114 papers in international conferences. He has served as Associate Editor for the IEEE Network, the IEEE Transactions on Systems, Man and Cybernetics – Part C and the IEEE Sensors Journal and the IEEE Communications Magazine.

Petros Nicoplitidis (SM'10, BS'98, PhD'02) is an Assistant Professor at the department of Informatics, Atirotole University of Thessaloniki. He has published more than 80 papers in international refereed journals and conferences. He is the coauthor of the book Wireless Networks (New York: Wiley, 2003). His research interests are in the areas of wireless networks and mobile communications.