Periodic Scheduling with Costs Revisited
A Novel Approach for Wireless Broadcasting

Christos Liaskos[1], Andreas Xeros[2], Georgios I. Papadimitriou[1], Marios Lestas[3], and Andreas Pitsillides[2]

[1] Dept. of Informatics, Aristotle University of Thessaloniki, 54124 Greece
{cliaskos,gp}@csd.auth.gr
[2] Dept.of Comp. Science, CY University, P.O. Box 20537 1678, Nicosia Cyprus
{csp6xa1,andreas.pitsillides}@cs.ucy.ac.cy
[3] Department of Electrical Engineering at Frederick University, Nicosia, Cyprus
eng.lm@frederick.ac.cy

Abstract. Periodic broadcast scheduling typically considers a set of discrete data items, characterized by their popularity, size and scheduling cost. A classic goal is the definition of an infinite, periodic schedule that yields minimum mean client serving time and minimum mean scheduling cost at the same time. This task has been proven to be NP-Hard and more recent works have discarded the scheduling cost attribute, focusing only on the minimization of the mean client serving time. In the context of the present work the scheduling cost is reinstated. An analysis-based scheduling technique is presented, which can practically minimize the mean client serving time and the mean scheduling cost concurrently. Comparison with related approaches yields superior performance in all test cases.

Keywords: periodic scheduling, broadcast cost, wireless transmission.

1 Introduction

Broadcasting is an efficient means of bandwidth preservation and information advertising in both wired and wireless environments [1, 2]. In the latter case, which is examined in the present work, broadcasting is much more vital, since there exists only one wireless medium. Thus, overcoming bandwidth limitations by adding more physical paths in not an option. The importance of broadcasting has highlighted the need for efficient broadcast scheduling, i.e. the proper serialization of data item transmission in order to ensure minimal client serving time and efficient quality of service.

The evaluation of wireless broadcast scheduling techniques takes place in a widely approved system setup [2–17]. A number of wireless clients freely roam an area covered by a broadcast network. All clients read the broadcasted data stream synchronously, while wireless transmission parameters are idealized in order to focus on the evaluation of the scheduling process only. The dataset to be broadcasted contains a number of discrete data items with known sizes. All the

Y. Koucheryavy et al. (Eds.): WWIC 2012, LNCS 7277, pp. 52–63, 2012.

aforementioned parameters may vary with time. Proposed scheduling techniques typically have an online expression [2–5], which can re-adapt the scheduling on the fly, based on new input regarding the client preferences or the update of the data set. The change in these parameters can be detected and handled by smart learning algorithms which are examined elsewhere, as a separate field of study [6, 18]. Two types of scheduling are defined: in pull-based scheduling the clients pose specific queries to a server [19]. The server then serializes the answers to the requests in a way that minimizes the mean serving time. In push-based (or periodic) scheduling, examined in the present study, the clients do not post queries to the server. The learning algorithm monitors the request probability of each available *data item* (or item class), typically through a lightweight, indirect feedback system [2]. (E.g. by exploiting Facebook profile data in a subscription-based system). Thus the actual number of users is not directly relevant [5]. The goal is then to create a periodic schedule that minimizes the mean serving time of the clients, based on the request probability distribution of the data items.

The problem with existing periodic scheduling approaches is that they do not take into account important, practical parameters, as for example copyright costs per item transmission or computational cost per item scheduling. A realistic scheduling authority will strive for a balance between client satisfaction and broadcast cost. This aspect of the scheduling process was originally taken into account in [9], where an additional attribute, the broadcast cost, was assigned to each data item. The set goal was the creation of the schedule that minimizes the mean client serving time and the mean scheduling cost at the same time. However, the problem was soon proven to be NP-Hard [7]. Subsequent studies validated the NP-Hardness of several variations of the original problem [8]. In the context of the studied push-based scheduling, the cost attribute was then discarded, and related works focused on the minimization of the mean serving time (or related metric) only [2, 4–6, 12–17]. In this case, the assumption of very large schedule sizes ("infinity" assumption) was adopted as a means of facilitating the mathematical analysis of the simplified problem [5]. This in turn resulted into unrealistically high computational requirements [4].

In the context of the present work we reinstate the broadcast cost attribute, and present a scheduler than can achieve minimal mean client serving time for any user-defined mean scheduling cost. Furthermore, it is proven that this task does not require infinite schedules, thus decreasing the overall computational complexity by orders of magnitude. The proposed scheduler in shown to be more efficient than the existing approaches, in all test cases. It is clarified that the proven theoretical NP-Hardness of the problem is not alleviated, but rather shown not to be prohibitive in practical communication systems.

1.1 Related Work

Research on push-based, wireless broadcast systems initially focused on the minimization of the clients' mean serving time, over an infinite time horizon, in the context of Teletext systems [10]. 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. [10] provided a solution, assuming equally-sized data items. The problem was revisited in [11], 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 same study proposed scheduling algorithms that achieved optimality at the expense of increased complexity ($O(N \cdot B)$, N being the number of data items and $B > N$ the number of scheduled broadcasts). These algorithms worked for small variations in item sizes only. Authors in [4] presented an analysis-derived periodic scheduler that achieved the same performance with $O(N)$ complexity, for any item sizes. In the same study it was also shown that the schedule size is a determining factor of the overall scheduling complexity. Simulations indicated that finite schedules may also achieve optimality.

Heuristic, low complexity scheduling methods were introduced in [3] 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. In [12], the authors applied clustering techniques for performing the data grouping. In [13] the grouping of items was analytically optimized. The analytical results were exploited in [14] for producing minimal complexity schedulers. All aforementioned studies focused on the minimization of the clients' mean serving time.

As previously stated, a more strict version of the scheduling problem assigns an additional attribute, the scheduling cost, to each data item. It is clarified that the scheduling cost is not related to broadcast deadlines, an extension of the mean serving time problem [15]. The new goal is to define the schedule that minimizes the mean serving time and the mean scheduling cost at the same time. The authors in [8] map the updated broadcast scheduling procedure to the generalized maintenance scheduling problem, which is a known NP-Hard problem. Several greedy algorithms are presented, as well as in [7], which generally operate beyond the analytically optimal bounds. To the best of the authors knowledge, no studies since [8] have attempted a practical solution for push-based systems, possibly due to the proof of NP-Hardness.

Standard Assumptions and Notation

We regard a set of N data items arbitrarily indexed by $i = 1...N$. Each item i is associated with its size l_i (in *bytes*) and its request probability p_i, $\sum_{i=1}^{N} p_i = 1$. Finally, $u_i \in \mathbb{N}^*$ will denote the number of occurrences of item i in the schedule.

No assumptions are made concerning the nature of a data item during the analysis. In accordance with the related work on scheduling, an item is simply a piece of information that a client may acquire through a single query [2–5, 7, 8, 10–17, 20, 21]. It is clarified that in push-based, periodic broadcast scheduling, the term "*client query*" does not imply posting a request to a server, but rather waiting for the broadcast of a specific item. According to [4], the mean serving

time achieved by a schedule is given by:

$$\overline{W} = \frac{1}{2} \cdot \left(\sum\nolimits_{i=1}^{N} u_i \cdot l_i \right) \cdot \left(\sum\nolimits_{i=1}^{N} \frac{p_i}{u_i} \right) \tag{1}$$

Notice that $\overline{W}$ does not depend on the number of clients, which are handled collectively as a Gaussian process via the central limit theorem [5]. In addition, (1) measures $\overline{W}$ in size units (e.g. bytes). Conversion to time units requires knowledge of the physical wireless transmission rate. The mean serving time is minimized when the item occurrences in the schedule follow the relation [11]:

$$u_i(\lambda) = \left[\left[\lambda \cdot \sqrt{\frac{p_i}{l_i}} \right] \right], i = 1 \ldots N, \ \lambda >> 1 \tag{2}$$

where $[\![.]\!]$ is the rounding function. Equation (2) is also known as *the square root rule*, and the condition $\lambda >> 1$ expresses the schedule size "infinity" assumption [5,7,8,10,11]. In order to avoid the nullification of the item occurrences, $u_i, \forall i$, it must generally hold that $\lambda \in \left(max \left\{ \frac{\sqrt{l_i}}{2 \cdot \sqrt{p_i}} \right\}, \infty \right)$. Finally, for a given λ, the total size of the schedule is expressed as:

$$L(\lambda) = \sum\nolimits_{i=1}^{N} u_i(\lambda) \cdot l_i \tag{3}$$

which is minimized when $u_i = 1, \forall i$.

The remainder of this paper is organized as follows: Section 2 presents the analysis leading to the definition of the novel, cost-aware optimal scheduler. Comparison with related work takes place in Section 3. Conclusion is given in Section 4.

2 Analysis of the Proposed Scheduling Scheme

We assign an additional, normalized attribute, $\alpha_i \in (0,1)$, to each of the data items $i = 1 \ldots N$. This attribute corresponds to the scheduling cost of [7,8], and is open to any physical interpretation that can be efficiently expressed in the value set $(0,1)$. Given a broadcast schedule of the available items, the mean cost is:

$$\overline{\alpha} = \frac{\sum_{i=1_i}^{N} u_i \cdot \alpha_i}{\sum_{i=1}^{N} u_i} \tag{4}$$

Notice that $\overline{\alpha}$ is minimized when $u_i = 0, \forall i \neq argin f_{(j)}\{a_j\}$, i.e. when we exclusively broadcast the item with the lowest cost. On the other hand, $\overline{W}$ is minimized when the relation (2) holds. Furthermore, the cost attributes, α_i, are not correlated in any way to the remaining item attributes, p_i, l_i. Therefore, minimizing $\overline{\alpha}$ and $\overline{W}$ concurrently requires the definition of a metric that combines both quantities. For example, [7] and [8] assume that both $\overline{\alpha}$ and $\overline{W}$ are of equal importance and define the combination:

$$S = 50\% \cdot \overline{\alpha} + 50\% \cdot \overline{W} \tag{5}$$

which needs to be minimized. However, the equal importance assumption is restrictive. In order to overcome this shortcoming, the following analysis will derive the full relation $\overline{W} = f(\overline{\alpha})$, i.e. the schedule that minimizes $\overline{W}$ for any given cost $\overline{\alpha}$. Any custom metric can then be satisfied at the intersection of the plot $\overline{W} = f(\overline{\alpha})$ and the line $\overline{W} = b \cdot \overline{\alpha}$, $b > 0$. As an example, the combination S of eq. (5) represents the very specific case of intersecting $\overline{W} = f(\overline{\alpha})$ and $\overline{W} = \overline{\alpha}$.

In order to enable the use of infinitesimal calculus, we will expand the value set of u_i from $\mathbb{N}$ to $\mathbb{R}_*^+$. Indeed, if u_i is extremely large $\forall i$, one can safely assume that $u_i \pm 0.5 \approx u_i$, where ± 0.5 represents any rounding error. Equation (4) then relates u_i to any arbitrary u_j, $j \in 1 \dots N$, $j \neq i$ as follows:

$$\frac{\partial u_j}{\partial u_i} = -\frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j} \tag{6}$$

Taking the first derivative of the mean client serving time, $\overline{W}$ (equation (1)), with regard to u_i, produces through (6):

$$\frac{\partial \overline{W}}{\partial u_i} = \frac{1}{2}\left[l_i - l_j \frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j}\right]_{A_1} \cdot \left(\sum_{k=1}^{N} \frac{p_k}{u_k}\right)_{B_1} + \dots$$
$$\dots + \frac{1}{2}\left(\sum_{k=1}^{N} u_k \cdot l_k\right)_{B_2} \cdot \left[-\frac{p_i}{u_i^2} + \frac{p_j}{u_j^2}\frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j}\right]_{A_2} \tag{7}$$

The reader may notice that the described procedure is an application of the Lagrange method of restricted optimization, on equations (1) and (4). The labels $A_{1,2}$, $B_{1,2}$ are added for quick referencing. Concerning the possible nullification of the $(\overline{\alpha} - \alpha_j)$ denominator, we simply note that α_j (i.e. reference item j) can be chosen to be different from the user-defined mean cost $\overline{\alpha}$, which is supplied as an input. We proceed to define Theorem 1:

Theorem 1. *Assume a large schedule (infinity assumption) and the request for minimal mean client serving time, $\overline{W}$, for a given mean scheduling cost, $\overline{\alpha}$. The corresponding optimal item occurrences are:*

$$u_i^{opt} = \left[\left[\lambda \cdot \sqrt{\frac{p_i}{l_j \cdot \frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j}}}\right]\right], \; j = argin f_{(i)} \left\{|\widetilde{\alpha} - \alpha_i|\right\}, \; \lambda >> 1 \tag{8}$$

where $\widetilde{\alpha}$ is the median of the $\{\alpha_i\}$ values.

Proof. We proceed to insert equation (8) in factor A_2 of (7). It is deduced after trivial calculations that $A_2 = 0$. Examining factor B_1 of equation (7) and statement $\lambda >> 1$ of (8), we derive that $B_1 \to 0$ due to the infinity assumption, while A_1 is finite. Therefore, the derivative of (7) is nullified, yielding optimality. This concludes the proof.

Remarks: The choice of the reference item j in equation (8) is not mandatory, but favors a restriction imposed by the square root of (8). i.e.:

$$l_i^{'} = l_j \cdot \frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j} > 0 \Longleftrightarrow \frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j} > 0, \forall i \tag{9}$$

which has the equivalent representation $(\overline{\alpha} - \alpha_i) \cdot (\overline{\alpha} - \alpha_j) > 0$. This restriction is not upheld when $\overline{\alpha}$ is inside the interval defined by α_j and α_i, $\forall i$. The choice of j as the item whose cost is nearest to the median, $\widetilde{\alpha}$, ensures that the interval between α_j and α_i, $\forall i$ is as small as possible, thus limiting possible issues.

However, the scheduling authority may require a mean cost $\overline{\alpha}$ that is indeed in the aforementioned interval. In this case we can exploit the fact that the infinity assumption makes for the existence of additional solutions. The reader may exemplary notice that inserting the transformation:

$$l_i^{'} = \left| l_j \cdot \frac{\overline{\alpha} - \alpha_i}{\overline{\alpha} - \alpha_j} \right| \tag{10}$$

in (8), also nullifies the derivative of (7), since the factors (B_1) and $(A_2 \cdot B_2)$ approach zero as λ (and therefore $u_i, \forall i$) increases. The drawback of this solution is that it requires much larger schedule sizes, since the factor $(A_2 \cdot B_2)$ must now also be nullified indirectly, by increasing the size of the schedule.

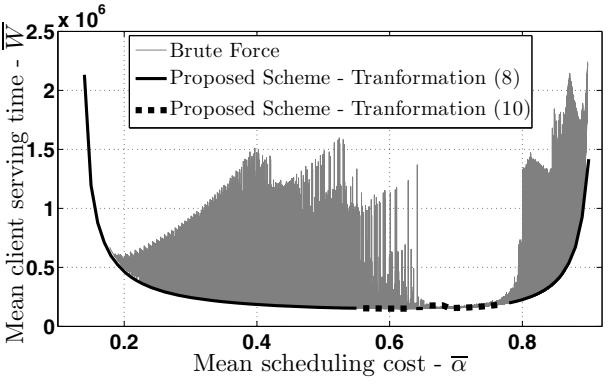

Fig. 1. Comparison of the proposed scheduler with results derived via brute force. For each given mean cost, $\overline{\alpha}$, the simple transformation of (9) produces a mean serving time, $\overline{W}$, that is optimal by brute-force standards. The dotted line corresponds to the use of transformation (10) in the cases where restrictions (9) do not apply. The mean serving time is measured in *Bytes*.

Figure 1 presents an indicative comparison of the proposed scheduling technique with the results derived via brute-force. We assume $N = 5$ items (a restriction enforced by the brute force procedure), with random sizes $l_{1-5} \in [10, 100] Kbyte$, random request probabilities, $\sum_{i=1}^{5} p_i = 1$, and random scheduling costs, $\alpha_{1-5} \in (0, 1)$. We simulate 1000 wireless clients listening to the same

broadcast schedule, each one waiting for the successive completion of 600 personal queries for items $1-5$. The creation of the queries obeys to the predefined, random request probabilities p_{1-5}. As in [2–17], we focus on the simulation of the scheduler. Therefore, it is assumed that there are no coverage, noise or interference issues, which could cause an altered perception of the efficiency of the proposed scheduler.

We seek a schedule with mean cost $\overline{\alpha} = 0.01 : 0.01 : 1$ and a minimum corresponding mean serving time, $\overline{W}(\overline{\alpha})$. For each $\overline{\alpha}$ value, we calculate the optimal number of item occurrences, u_i, through equation (8). Transformation (10) is employed only when (8) fails to comply with (9). Once the optimal u_i have been calculated, we can use any serialization technique which targets the creation of periodic schedules. In the case of Fig. 1 we adopt the serialization scheme of [4]. This serializer receives the desired item occurrences, u_i, and the item sizes, l_i, as input, and produces a periodic schedule by employing preemption. In the case of brute force we use the same serialization scheme, but we try all possible $u_{1-5} = 1 : 1 : 500$ combinations. Then, for each achieved mean scheduling cost, we log the smallest achieved mean serving time. The results of Fig. 1 yield convergence between the proposed scheme and the brute-force approach. Small discrepancies of the brute-force results are attributed to the computational limitations of the procedure.

The almost perfect results presented in Fig. 1 do not in any way falsify the proof of the NP-Hardness of the scheduling problem [8]. They do however pose a question of whether purely theoretical assumptions have magnified the practical significance of the issue. It has been proven that an optimal schedule is periodic: the interval between two consecutive occurrences of an item must be constant [10]. Therefore, knowledge of the u_i ratio is at first sufficient for creating an optimal schedule. However, large variations in the size of the data items may hinder periodicity. (E.g. a huge file may not fit in its predefined, periodic positions) [4]. The NP-Hardness stems from the resulting combinatorial problem of finding the optimal item serialization in this case. This issue is not a practical hindrance however: in the vast majority of the modern communication systems, large files are divided in much smaller segments or packets, prior to transmission. This approach is known to resolve the aforementioned issue in periodic scheduling as well [4]. For limited item size variations, non-preemptive serializers (e.g. [5]) produced identical, optimal results. Conclusively, having shown that the definition of the optimal u_i ratios is easily tractable, the authors claim that the undisputed, theoretical NP-Hardness of the scheduling problem does not pose a significant limitation in practice.

3 Comparison with Related Work

The proposed scheduler is compared with related approaches in terms of:

- **Tunability**. A scheduler should be able to operate at any $\{\overline{W}, \overline{\alpha}\}$ (mean serving time, mean cost) combination dictated by the scheduling authority.

- **Efficiency**. The scheduler should achieve minimal mean client serving time, $\overline{W}$, for any selected mean scheduling cost, $\overline{\alpha}$.
- **Complexity**. A scheduler should require the least possible computational resources for the production of the chosen schedule. As proven in [4,14], the size of a schedule defines the overall complexity and, therefore, should be kept minimal.

The proposed scheduler is compared to the *Randomized*, *Greedy* and *Periodic* schedulers of [7,8]. These schedulers are cost-aware, assigning a single attribute α_i per data item as well, and represented the most viable options prior to the present work. As a reference point, we include the cost-agnostic schedulers of [5] and [4], which target solely the minimization of the mean client serving time, disregarding the cost. The scheduler of [4] surpassed [5] in terms of complexity and performance, and represents a state-of-the-art (SOA) algorithm for periodic scheduling. Notice that [4] comprises a scheduling technique, and a serialization technique. The latter is generic and reusable, as discussed in the context of Fig. 1. We use this serialization scheme in all applicable compared approaches for fairness reasons. However, each compared approach has its own scheduling scheme, i.e. a way of setting the optimal item occurrences, u_i.

We assume the same simulation configuration employed previously, in the experiments of Fig. 1. The number of data items is raised to $N = 50$, and their request probabilities are set by the ZIPF distribution with a skewness factor of 0.6 [22]. Their sizes are picked randomly in $[10, 100]KBytes$, and theirs costs in $(0, 1)$. The topology, the number of clients and the number of queries remain unaltered.

We examine the values $\overline{\alpha} = [0.01 : 0.01 : 1]$ as possible requests for the mean scheduling cost. For each value we require from the compared schedulers to create corresponding optimal schedules, and we log the achieved mean client serving times (MST) and schedule sizes. The results are shown in Fig. 2 and 3.

The proposed scheduler can produce multiple optimal schedules per $\overline{\alpha}$ choice. This is due to the fact that the serialization technique of [4] can also fine-tune the size of the produced schedule, with trivial impact on the efficiency (smaller schedules-slightly higher mean serving time/cost). This phenomenon creates the grayed surface of possible operation points for the proposed solution in Fig 2 and 3. In terms of tunability, the proposed scheduler flawlessly covers the complete range of tested $\overline{\alpha}$ values. All other algorithms however present zero tunability, being able to operate only at one, not user defined mean cost. This is not surprising for the cost agnostic algorithms, but is not in favor of the cost-aware ones. Furthermore, the cost-awareness of the latter ones does not have a significant overall impact, as their operation points nearly coincide with the those of the cost-agnostic scheduler of [5]. This issue is more observable in the top view of Fig. 3.

The studies of [7,8], where the related cost-aware schedulers are proposed, do not target the minimization of the mean client serving time for a given cost. Instead, both studies seek to minimize the sum $S = 50\% \cdot \overline{\alpha} + 50\% \cdot \overline{W}$ of equation (5). This results into only one possible operation point and zero tunability.

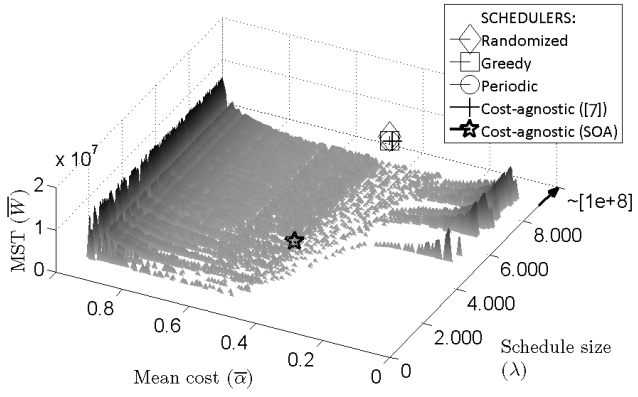

Fig. 2. Tunability of the compared schedulers in terms of possible operation points, $\{\overline{\alpha}, \overline{W}, \lambda\}$. The proposed scheduler can efficiently cover all requests for scheduling cost $\overline{\alpha} = [0.01 : 0.01 : 1]$, enabling application-specific, fine grained balancing between performance and cost (grayed plane). The related, cost-aware approaches are limited to one, not-user defined point of operation, with deterring corresponding schedule size. The arrow corresponds to a significant leap in the y-axis for presentational purposes.

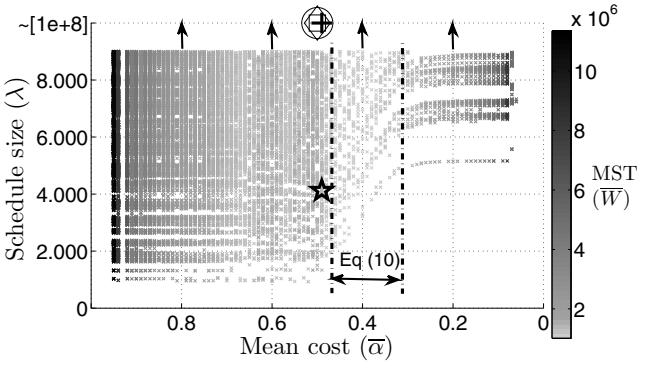

Fig. 3. Top view of Fig. 2. Even though cost-aware, the Randomized, Periodic and Greedy schedulers do not offer significant advantages over the cost-agnostic approaches. In fact, SOA may be a better choice instead, because of the decreased schedule size. The use of transformation (10) causes increase in the size of the schedules produced by the proposed scheduler, as expected by the analysis. Notice that the arrows correspond to a significant leap in the y-axis for presentational purposes.

Furthermore, with the NP-Hardness of the problem a given, these studies mainly target the proposal of a very lax (and therefore suboptimal) but safely achievable lower bound of the S quantity. Consequently, the *Randomized*, *Greedy* and *Periodic* schedulers presented therein do not behave significantly better than the cost-agnostic approaches [4, 5].

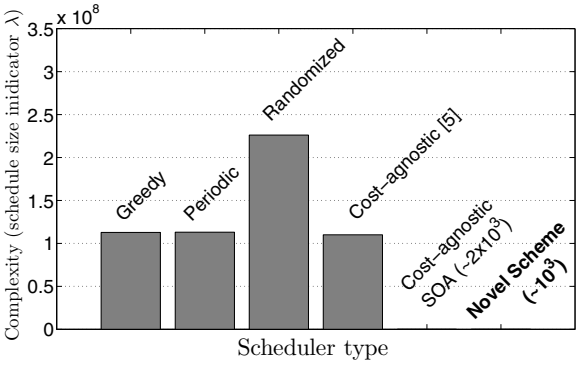

Fig. 4. Required schedule sizes, in the case of $\overline{\alpha} = 0.47$. As shown in Fig. 3, this case represents the sole possible operation point of the compared, related schedulers. The novel scheme requires $\sim 10^5$ times less computational resources. Randomized item serialization hurts periodicity and, utterly, performance.

As an additional remark on the results of Fig. 3, notice that the region between the vertical, dotted lines designates the use of transformation (10) by the proposed scheduler. According to the analysis of Section 2, the use of this transformation yields optimality at the expense of increased schedule size. The results concur to this claim and the possible operation points are sparser inside the designated region as well.

Concerning the size of the produced schedules (and therefore the complexity of the corresponding schedulers), the novel scheduler achieves smaller sizes by 5 orders of magnitude. This fact is observable in Fig. 2 and 3, clarifying that the black arrows represent a leap in λ values by five orders of magnitude ($\lambda = 9000 \rightarrow \lambda \approx 10^8$), for presentational purposes. The difference in complexity is better illustrated in Fig. 4. The figure presents the required schedule sizes in the case of $\overline{\alpha} = 0.47$, i.e. the only possible operation point for the related schedulers. It is evident that the computational requirements for the scheduling procedure can be reduced by a factor of 10^5, without significant impact on performance. The randomized scheduler of [8] produces the worst results, since random item serialization may hurt periodicity, which is a prerequisite for optimality [10].

4 Conclusion

The present study reinstated the broadcast cost per data item as a vital factor of the periodic scheduling process. A novel scheduler was proposed which can achieve minimal client serving times for any requested mean scheduling cost. The computational requirements of the scheduler were decreased by reducing the size of the produced schedule. Conclusively, the study demonstrated that the theoretically NP-Hard problem of periodic, push-based broadcast scheduling with costs may have an efficient solution in practice. Comparison with related approaches yielded improved performance, combined with fine-grained operational tunability.

Acknowledgment. This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heracleitus II. Investing in knowledge society through the European Social Fund.

References

1. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M.F., Briggs, N.H., Braynard, R.L.: Networking named content. In: Proceedings of the 5th ACM Conference on Emerging Network Experiment and Technology (CoNEXT 2009), Rome, Italy, pp. 1–12 (December 2009)
2. Nicopolitidis, P., Papadimitriou, G., Pomportsis, A.: Continuous Flow Wireless Data Broadcasting for High-Speed Environments. IEEE Transactions on Broadcasting 55(2), 260–269 (2009)
3. Acharya, S., Alonso, R., Franklin, M., Zdonik, S.: Broadcast disks. ACM SIGMOD Record 24(2), 199–210 (1995)
4. Liaskos, C., Petridou, S., Papadimitriou, G.: Towards Realizable, Low-Cost Broadcast Systems for Dynamic Environments. IEEE/ACM Transactions on Networking 19(2), 383–392 (2011)
5. Vaidya, N.H., Hameed, S.: Scheduling data broadcast in asymmetric communication environments. Wireless Networks 5(3), 171–182 (1999)
6. Kakali, V.L., Sarigiannidis, P.G., Papadimitriou, G.I., Pomportsis, A.S.: A Novel Adaptive Framework for Wireless Push Systems Based on Distributed Learning Automata. Wireless Personal Communications 57(4), 591–606 (2011)
7. Schabanel, N.: The Data Broadcast Problem with Preemption. In: Reichel, H., Tison, S. (eds.) STACS 2000. LNCS, vol. 1770, pp. 181–192. Springer, Heidelberg (2000)
8. Kenyon, C., Schabanel, N.: The Data Broadcast Problem with Non-Uniform Transmission Times. Algorithmica 35(2), 146–175 (2002)
9. Gondhalekar, V., Jain, R., Werth, J.: Scheduling on airdisks: efficient access to personalized information services via periodic wireless data broadcast. In: Proceedings of the 1997 International Conference on Communications (ICC 1997), Montreal, Canada, pp. 1276–1280. IEEE (June 1997)
10. Gecsei, J.: The Architecture of Videotex Systems. Prentice-Hall, Englewood Cliffs (1983)
11. Su, C.J., Tassiulas, L.: Broadcast scheduling for information distribution. In: Proceedings of the 17th Annual IEEE Conference on Computer Communications, INFOCOM 1997, Kobe, Japan, pp. 109–117. IEEE Comput. Soc. Press (April 1997)
12. Liaskos, C., Petridou, S., Papadimitriou, G., Nicopolitidis, P., Obaidat, M., Pomportsis, A.: Clustering-driven Wireless Data Broadcasting. IEEE Wireless Comm. Magazine 16, 80–87 (2009)
13. Liaskos, C., Petridou, S., Papadimitriou, G., Nicopolitidis, P., Pomportsis, A.: On the Analytical Performance Optimization of Wireless Data Broadcasting. IEEE Transactions on Vehicular Technology 59, 884–895 (2010)
14. Liaskos, C., Petridou, S., Papadimitriou, G.: Cost-Aware Wireless Data Broadcasting. IEEE Transactions on Broadcasting 56(1), 66–76 (2010)
15. Xu, J., Tang, X., Lee, W.-C.: Time-critical on-demand data broadcast: algorithms, analysis, and performance evaluation. IEEE Transactions on Parallel and Distributed Systems 17(1), 3–14 (2006)

16. Zheng, B., Wu, X., Jin, X., Lee, D.L.: TOSA: a near-optimal scheduling algorithm for multi-channel data broadcast. In: Proceedings of the 6th International Conference on Mobile Data Management (MDM 2005), Ayia Napa, Cyprus, pp. 29–37 (May 2005)
17. Hu, C.-L., Chen, M.-S.: Online Scheduling Sequential Objects with Periodicity for Dynamic Information Dissemination. IEEE Transactions on Knowledge and Data Engineering 21(2), 273–286 (2009)
18. Poor, H.V., Hadjiliadis, O.: Quickest detection. Cambridge University Press, Cambridge (2009), http://www.worldcat.org/oclc/637444316
19. Bansal, N., Coppersmith, D., Sviridenko, M.: Improved Approximation Algorithms for Broadcast Scheduling. SIAM Journal on Computing 38(3), 1157 (2008)
20. Xu, J., Lee, D.-L., Hu, Q., Lee, W.-C.: Data Broadcast. In: Stojmenović, I., Zomaya, A.Y. (eds.) Wiley Series on Parallel and Distributed Computing, pp. 243–265. John Wiley & Sons, Inc., New York (2002)
21. Chen, M.-S., Wu, K.-L., Yu, P.: Optimizing index allocation for sequential data broadcasting in wireless mobile computing. IEEE Transactions on Knowledge and Data Engineering 15(1), 161–173 (2003)
22. Pietronero, L., Tosatti, E., Tosatti, V., Vespignani, A.: Explaining the uneven distribution of numbers in nature: The laws of Benford and Zipf. Physica A: Statistical Mechanics and its Applications 293(1-2), 297–304 (2001)