

Optimal Backlog in the Plane

Valentin Polishchuk and Jukka Suomela

Helsinki Institute for Information Technology HIIT
Helsinki University of Technology and University of Helsinki
P.O. Box 68, FI-00014 University of Helsinki, Finland
`valentin.polishchuk@cs.helsinki.fi`, `jukka.suomela@cs.helsinki.fi`

Abstract. Suppose that a cup is installed at every point of a planar set P , and that somebody pours water into the cups. The total rate at which the water flows into the cups is 1. A player moves in the plane with unit speed, emptying the cups. At any time, the player sees how much water there is in every cup. The player has no information on how the water will be poured into the cups in the future; in particular, the pouring may depend on the player's motion. The *backlog* of the player is the maximum amount of water in any cup at any time, and the player's objective is to minimise the backlog. Let D be the diameter of P . If the water is poured at the rate of $1/2$ into the cups at the ends of a diameter, the backlog is $\Omega(D)$. We show that there is a strategy for the player that guarantees the backlog of $O(D)$, matching the lower bound up to a multiplicative constant. Note that our guarantee is independent of the number of the cups.

1 Introduction

Consider a wireless sensor network where each sensor node stores data locally on a flash memory card [1–5]. To conserve energy, radio communication is only used for control information; for example, each network node periodically reports the amount of data stored on the memory card. We have one maintenance man who visits the sensors and physically gathers the information stored on the memory cards. The maximum amount of data produced at a sensor between consecutive visits – the *backlog* – determines how large memory cards are needed. In this work we study the problem of designing a route for the maintenance man to minimise the backlog.

If each device produces data at the same constant rate, then we have an offline optimisation problem which is exactly the travelling salesman problem: find a minimum-length tour that visits each device.

However, in the general case, the data rate may vary from time to time and from device to device; for example, a motion-tracking application produces data only when there is a monitored entity in the immediate neighbourhood of the sensor. We arrive at an *online* problem: guide the maintenance man so that the maximum backlog is minimised, using only the information on the current backlog.

We only assume that we know the *total* volume of the data that is gathered on the memory cards in one time unit; we assume nothing about the distribution of the data. For example, our network could be tracking a constant number of moving objects, and each of the objects produces data at a constant rate at its nearest sensor; we assume nothing about how the objects move. At first glance assuming so little may seem overly cautious, but as we shall see, we can nevertheless obtain strong positive results.

Our primary interest is in the scalability of the system. Trivially, doubling the total rate of the data means that the backlog may double as well. We focus on the effect of increasing the number of sensor devices and increasing the physical area within which the devices are installed.

Surprisingly, it turns out that if the diameter of the physical area is fixed and the total rate at which the data accumulates at the devices is fixed, we can guarantee a certain amount of backlog *regardless of the number or placement* of the sensors. The adversary can install any number of sensors in arbitrary locations, and the adversary can also change the distribution of the data depending on the activities of our maintenance man; nevertheless, we can keep the backlog below a certain level.

As a direct consequence, the backlog increases only linearly in the diameter of the physical area. If the diameter is doubled, we only need to double the speed of our maintenance man – or double the size of the memory cards.

1.1 Problem Formulation and Contribution

Formally, the *minimum backlog game* is defined as follows. Because of historical precedent [6, 7], we use cups and water instead of sensor devices and data.

Definition 1 (Minimum backlog game). *Let $P \subset \mathbb{R}^2$ be a finite planar set. There is a cup on each point in P . The adversary pours water into the cups P ; the total rate at which the water is poured into the cups is 1. The player moves in the plane with unit speed, starting at an arbitrary point. Whenever the player visits a cup, the cup is emptied. The goal of the player is to keep the maximum level of water in any cup at any time as low as possible.*

Let D be the diameter of P . Clearly, the adversary can ensure that some cup at some time will contain $\Omega(D)$ water: the adversary may just take two cups that define the diameter of P , and pour water with the rate $1/2$ into each of them. In this paper, we prove that the player can match this, up to a multiplicative constant.

Theorem 1. *The player has a strategy which guarantees that no cup ever contains more than $O(D)$ units of water.*

We emphasise that the result does not depend on the number of cups in the set, but just on its diameter.

1.2 Comparison with Previous Work

In the *discrete* version of the game [7], the cups are installed at the nodes of a graph. The game proceeds in *time steps*; in each step the adversary pours a total of 1 water, while the player moves from a node to an adjacent node. The competitive ratio of any algorithm has a lower bound of $\Omega(\Delta)$, where Δ is the diameter of the graph [7]. This motivated determining a strategy for the player that guarantees a uniform upper bound on the backlog.

In an arbitrary graph on n nodes, the deamortization analysis of Dietz and Sleator [8] leads to a strategy that guarantees $O(\Delta \log n)$ backlog. In certain graphs (stars) the adversary can guarantee a matching $\Omega(\Delta \log n)$ lower bound. The main contribution of Bender et al. [7] was an algorithm achieving a backlog of $O(n\sqrt{\log \log n})$ in the case when the graph is an n -by- n grid. It was sketched how to extend the regular-grid approach to the game on a general planar set P to guarantee a backlog of $O(D\sqrt{\log \log |P|})$, where D is the diameter of P .

Our strategy guarantees a backlog of $O(D)$, independent of the number of cups. The strategy consists of a set of coroutines, each responsible for clearing water of certain age. Similarly to prior work [7], we compare the performance of the coroutines to that of a player in an “imaginary” game. We give bounds on the performance of each of the coroutines, combining them into the bound for the topmost algorithm.

2 Preliminaries

We define the concept of a (τ, k) -game, forming the basis of our analysis. We give a preliminary lemma about the performance of the player in the game; the lemma is a direct extension of a result of Dietz and Sleator [8]. We also recall a result of Few [9] stating that for any planar set Q there exists a cycle through Q of length $O(\text{diam}(Q)\sqrt{|Q|})$.

2.1 The (τ, k) -game

For each $\tau \in \mathbb{R}$, $k \in \mathbb{N}$ we define the (τ, k) -game as follows.

Definition 2. *There is a set of cups, initially empty, not located in any particular metric space. At each time step the following takes place, in this order:*

1. *The adversary pours a total of τ units of water into the cups. The adversary is free to distribute the water in any way he likes.*
2. *The player empties k fullest cups.*

The game is discrete – the player and the adversary take turns making moves during discrete time steps. The next lemma bounds the amount of water in any cup after few steps of the game.

Lemma 1. *The water level in any cup after r complete time steps of the (τ, k) -game is at most $H_r\tau/k$, where H_r is the r th harmonic number.*

Proof. We follow the analysis of Dietz and Sleator [8, Theorem 5]. Consider the water levels in the cups after the time step j . Let $X_j^{(i)}$ be the amount of water in the cup that is i th fullest, and let

$$S_j = \sum_{i=1}^{(r-j)k+1} X_j^{(i)}$$

be the total amount of water in $(r-j)k+1$ fullest cups at that time. Initially, $j=0$, $X_0^{(i)}=0$, and $S_0=0$.

Let us consider what happens during the time step $j \in \{1, 2, \dots, r\}$. The adversary pours τ units of water; the total amount of water in $(r-j+1)k+1$ fullest cups is therefore at most $S_{j-1} + \tau$ after the adversary's move and before the player's move (the worst case being that the adversary pours all water into $(r-j+1)k+1$ fullest cups).

Then the player empties k cups. The k fullest cups contained at least a fraction $k/((r-j+1)k+1)$ of all water in the $(r-j+1)k+1$ fullest cups; the remaining $(r-j)k+1$ cups are now the fullest. We obtain the inequality

$$S_j \leq \left(1 - \frac{k}{(r-j+1)k+1}\right) (\tau + S_{j-1})$$

or

$$\frac{S_j}{(r-j)k+1} \leq \frac{\tau}{(r-j+1)k+1} + \frac{S_{j-1}}{(r-j+1)k+1}.$$

Therefore the fullest cup after time step r has the water level at most

$$\begin{aligned} X_r^{(1)} &= \frac{S_r}{k(r-r)+1} \\ &\leq \frac{\tau}{1k+1} + \frac{\tau}{2k+1} + \dots + \frac{\tau}{rk+1} + \frac{S_0}{rk+1} \\ &\leq \frac{\tau}{k} \left(\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{r} \right). \quad \square \end{aligned}$$

2.2 Few's Lemma

Our strategy for the minimum backlog game invokes a number of coroutines at different moments of time. The following result by Few [9] provides the basis of the proof that the execution of the coroutines can indeed be scheduled as we define.

Lemma 2. [9, Theorem 1] *Given n points in a unit square, there is a path through the n points of length not exceeding $\sqrt{2n} + 1.75$.*

We make use of the following corollary.

Corollary 1. *Let S be a $D \times D$ square. Let $i \in \{0, 1, \dots\}$. Let $Q \subset S$ be a planar point set with $|Q| = 25^i$ and $\text{diam}(Q) = D$. For any point $p \in S$ there exists a closed tour of length at most $5^{i+1}D$ that starts at p , visits all points in Q , and returns to p .*

Proof. If $i > 0$, by Lemma 2, there is tour of length at most

$$(\sqrt{2(25^i + 1)} + 1.75 + \sqrt{2})D \leq 5^{i+1}D$$

that starts at p , visits all points in Q , and returns to p . If $i = 0$, there is a tour of length $2\sqrt{2}D \leq 5D$ through p and $|Q| = 1$ points. \square

3 The Strategy

The player's strategy is composed from a number of coroutines, which we label with $i \in \{0, 1, \dots\}$. The coroutine i is invoked at times $(10L + \ell)\tau_i$ for each $L \in \{0, 1, \dots\}$ and $\ell \in \{1, 2, \dots, 10\}$. Whenever a lower-numbered coroutine is invoked, higher-numbered coroutines are suspended until the lower-numbered coroutine returns.

We choose the values τ_i as follows. For $i \in \{0, 1, 2, \dots\}$, let

$$\begin{aligned} k_i &= 25^i, \\ \tau_i &= (2/5)^i \cdot 10Dk_i = 10^i \cdot 10D. \end{aligned}$$

For $L \in \{0, 1, \dots\}$ and $\ell \in \{1, 2, \dots, 10\}$, define (i, L, ℓ) -*water* to be the water that was poured during the time interval $[10L\tau_i, (10L + \ell)\tau_i]$.

3.1 The Coroutine i

The coroutine i performs the following tasks when invoked at time $(10L + \ell)\tau_i$:

1. Determine which k_i cups to empty. The coroutine chooses to empty k_i cups with the largest amount of (i, L, ℓ) -water.
2. Choose a cycle of length at most $\tau_i/2^{i+1}$ which visits the k_i cups and returns back to the original position. This is possible by Corollary 1 because $5^{i+1}D = \tau_i/2^{i+1}$.
3. Guide the player through the chosen cycle.
4. Return.

Observe that when a coroutine returns, the player is back in the location from which the cycle started. Therefore invocations of lower-numbered coroutines do not interfere with any higher-number coroutines which are currently suspended; they just delay the completion of the higher-numbered coroutines.

The completion is not delayed for too long. Indeed, consider a time period $[j\tau_i, (j+1)\tau_i]$ between consecutive invocations of the coroutine i . For $h \in \{0, 1, \dots, i\}$, the coroutine h is invoked 10^{i-h} times during the period. The cycles of the coroutine h have total length at most

$$10^{i-h}\tau_h/2^{h+1} = \tau_i/2^{h+1}.$$

In grand total, all coroutines $0, 1, \dots, i$ invoked during the time period take time

$$\sum_{h=0}^i \tau_i/2^{h+1} < \tau_i.$$

Therefore all coroutines invoked during the time period are able to complete within it. This proves that the execution of the coroutines can be scheduled as described.

4 Analysis

We now analyse the backlog under the above strategy. For any points in time $0 \leq t_1 \leq t_2 \leq t_3$, we write $W([t_1, t_2], t_3)$ for the maximum per-cup amount of water that was poured during the time interval $[t_1, t_2]$ and is still in the cups at time t_3 .

We need to show that $W([0, t], t)$ is bounded by a constant that does not depend on t . To this end, we first bound the amount of (i, L, ℓ) -water present in the cups at the time $(10L + \ell + 1)\tau_i$. Then we bound the maximum per-cup amount of water at an arbitrary moment of time by decomposing the water into (i, L, ℓ) -waters and a small remainder.

We make use of the following simple properties of $W([\cdot, \cdot], \cdot)$. Consider any four points in time $0 \leq t_1 \leq t_2 \leq t_3 \leq t_4$. First, the backlog for *old* water is nonincreasing: $W([t_1, t_2], t_4) \leq W([t_1, t_2], t_3)$. Second, we can decompose the backlog into smaller parts: $W([t_1, t_3], t_4) \leq W([t_1, t_2], t_4) + W([t_2, t_3], t_4)$.

4.1 (i, L, ℓ) -Water at Time $(10L + \ell + 1)\tau_i$

Let $i \in \{0, 1, \dots\}$, $L \in \{0, 1, \dots\}$, and $\ell \in \{1, 2, \dots, 10\}$. Consider (i, L, ℓ) -water and the activities of the coroutine i when it was invoked at the times $(10L + 1)\tau_i$, $(10L + 2)\tau_i, \dots, (10L + \ell)\tau_i$.

The crucial observation is the following. By the time $(10L + \ell + 1)\tau_i$, the coroutine i has, in essence, played a (τ_i, k_i) -game for ℓ rounds with (i, L, ℓ) -water. The difference is that the coroutine cannot empty the cups immediately after the adversary's move; instead, the cup-emptying takes place during the adversary's next move. Temporarily, some cups may be fuller than in the (τ_i, k_i) -game. However, once we wait for τ_i time units to let the coroutine i complete its clean-up tour, the maximum level of the water that has arrived before the beginning of the clean-up tour is at most that in the (τ_i, k_i) -game.

The fact that the emptying of the cups is delayed can only hurt the adversary, as during the clean-up tour the player may also accidentally undo some of the cup-filling that the adversary has performed on his turn. The same applies to the intervening lower-numbered coroutines.

Therefore, by Lemma 1, we have

$$W([10L\tau_i, (10L + \ell)\tau_i], (10L + \ell + 1)\tau_i) \leq H_\ell \cdot \tau_i/k_i < 3\tau_i/k_i. \quad (1)$$

4.2 Decomposing Arbitrary Time t

Let t be an arbitrary instant of time. We can write t as $t = T\tau_0 + \epsilon$ for a nonnegative integer T and some remainder $0 \leq \epsilon < \tau_0$. Furthermore, we can

represent the integer T as

$$T = \ell_0 + 10\ell_1 + \cdots + 10^N \ell_N$$

for some integers $N \in \{0, 1, \dots\}$ and $\ell_i \in \{1, 2, \dots, 10\}$. Since the range is $1 \leq \ell_i \leq 10$, not $0 \leq \ell_i \leq 9$, this is not quite the usual decimal representation; we chose this range for ℓ_i to make sure that ℓ_i is never equal to 0.

We also need partial sums

$$L_i = \ell_{i+1} + 10\ell_{i+2} + \cdots + 10^{N-i-1}\ell_N.$$

Put otherwise, for each $i \in \{0, 1, \dots, N\}$ we have

$$T = \ell_0 + 10\ell_1 + \cdots + 10^i \ell_i + 10^{i+1} L_i$$

and therefore

$$\begin{aligned} t &= \epsilon + \ell_0 \tau_0 + \ell_1 \tau_1 + \cdots + \ell_N \tau_N \\ &= \epsilon + \ell_0 \tau_0 + \ell_1 \tau_1 + \cdots + \ell_i \tau_i + 10 L_i \tau_i. \end{aligned}$$

We partition the time from 0 to $t - \epsilon$ into long and short periods. The *long period* $i \in \{0, 1, \dots, N\}$ is of the form $[10L_i \tau_i, (10L_i + \ell_i - 1)\tau_i]$ and the *short period* i is of the form $[(10L_i + \ell_i - 1)\tau_i, (10L_i + \ell_i)\tau_i]$. See Fig. 1 for an illustration. Short periods are always nonempty, but the long period i is empty if $\ell_i = 1$.

4.3 Any Water at Arbitrary Time t

Now we make use of the decomposition of an arbitrary time interval $[0, t]$ defined in the previous section: we have long periods $i \in \{0, 1, \dots, N\}$, short periods $i \in \{0, 1, \dots, N\}$, and the remainder $[t - \epsilon, t]$.

Consider the long period i . We bound the backlog from this period by considering the point in time $(10L_i + \ell_i)\tau_i \leq t$. If the period is nonempty, that is, $\ell_i > 1$, then we have by (1)

$$\begin{aligned} &W([10L_i \tau_i, (10L_i + \ell_i - 1)\tau_i], t) \\ &\leq W([10L_i \tau_i, (10L_i + \ell_i - 1)\tau_i], (10L_i + \ell_i)\tau_i) < 3\tau_i/k_i. \end{aligned} \quad (2)$$

Naturally, if the period is empty, then (2) holds as well.

Consider a short period i for $i > 0$. It will be more convenient to write the short period in the form $[10L'_i \tau_{i-1}, (10L'_i + 10)\tau_{i-1}]$ where $L'_i = 10L_i + \ell_i - 1$ is a nonnegative integer (this is illustrated in Fig. 1 for $i = 3$). We bound the backlog from this period by considering the point in time $(10L'_i + 11)\tau_{i-1} \leq t$. By (1) we have

$$\begin{aligned} &W([(10L_i + \ell_i - 1)\tau_i, (10L_i + \ell_i)\tau_i], t) \\ &= W([10L'_i \tau_{i-1}, (10L'_i + 10)\tau_{i-1}], t) \\ &\leq W([10L'_i \tau_{i-1}, (10L'_i + 10)\tau_{i-1}], (10L_i + 11)\tau_{i-1}) < 3\tau_{i-1}/k_{i-1}. \end{aligned} \quad (3)$$

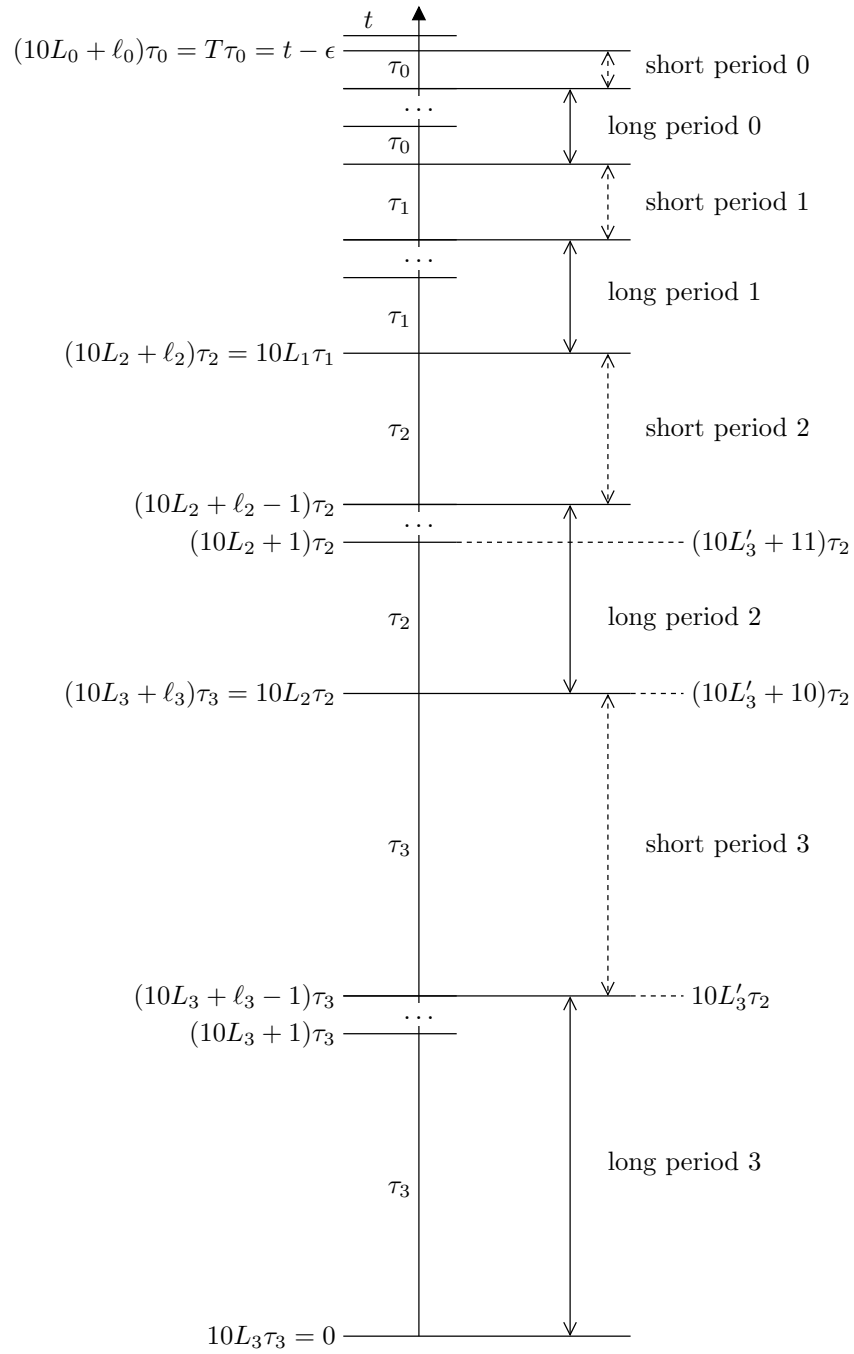


Fig. 1. Decomposition of the time; in this example, $N = 3$. The illustration is not in scale; actually $\tau_i = 10\tau_{i-1}$.

Next consider the short period $i = 0$. We have the trivial bound τ_0 for the water that arrived during the period. Therefore

$$W\left([(10L_0 + \ell_0 - 1)\tau_0, (10L_0 + \ell_0)\tau_0], t\right) \leq \tau_0. \quad (4)$$

Finally, we have the time segment from $t - \epsilon$ to t . Again, we have the trivial bound ϵ for the water that arrived during the time segment. Therefore

$$W\left([t - \epsilon, t], t\right) \leq \epsilon < \tau_0. \quad (5)$$

Now we can obtain an upper bound for the backlog at time t . Summing up (2), (3), (4), and (5), we have the maximum backlog

$$\begin{aligned} W([0, t], t) &\leq \sum_{i=0}^N W\left([10L_i\tau_i, (10L_i + \ell_i - 1)\tau_i], t\right) \\ &\quad + \sum_{i=0}^N W\left([(10L_i + \ell_i - 1)\tau_i, (10L_i + \ell_i)\tau_i], t\right) \\ &\quad + W\left([t - \epsilon, t], t\right) \\ &\leq \sum_{i=0}^N 3\tau_i/k_i + \sum_{i=1}^N 3\tau_{i-1}/k_{i-1} + 2\tau_0 \\ &\leq \sum_{i=0}^{\infty} 6\tau_i/k_i + 2\tau_0 \\ &= 60D \sum_{i=0}^{\infty} (2/5)^i + 20D = 120D = O(D). \end{aligned}$$

5 Conclusions

We have shown that the player in the minimum backlog game has a strategy where the backlog does not depend on the number of cups, but only on the diameter of the cups set. This implies that the backlog scales linearly with the diameter of the area.

An interesting open question is the scalability in the *number of players*. If we have four players instead of one, we can divide the area into four parts and assign each player into one of the parts; this effectively halves the diameter and thus halves the backlog. It remains to be investigated whether we can exploit multiple players in a more efficient manner.

Acknowledgements

We thank Esther Arkin, Michael Bender, Sándor Fekete, Alexander Kröller, Vincenzo Liberatore, and Joseph Mitchell for discussions. This research was supported in part by the Academy of Finland, Grants 116547 and 118653 (ALGO-DAN), and by Helsinki Graduate School in Computer Science and Engineering (Hecse).

References

1. Somasundara, A.A., Ramamoorthy, A., Srivastava, M.B.: Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In: Proc. 25th IEEE International Real-Time Systems Symposium (RTSS, Lisbon, Portugal, December 2004), Los Alamitos, CA, USA, IEEE Computer Society Press (2004) 296–305
2. Jea, D., Somasundara, A., Srivastava, M.: Multiple controlled mobile elements (data mules) for data collection in sensor networks. In: Proc. 1st IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS, Marina del Rey, CA, USA, June–July 2005). Volume 3560 of Lecture Notes in Computer Science, Berlin, Germany, Springer-Verlag (2005) 244–257
3. Gu, Y., Bozdağ, D., Brewer, R.W., Ekici, E.: Data harvesting with mobile elements in wireless sensor networks. *Computer Networks* **50**(17) (2006) 3449–3465
4. Mathur, G., Desnoyers, P., Ganesan, D., Shenoy, P.: Ultra-low power data storage for sensor networks. In: Proc. 5th International Conference on Information Processing in Sensor Networks (IPSN, Nashville, TN, USA, April 2006), New York, NY, USA, ACM Press (2006) 374–381
5. Diao, Y., Ganesan, D., Mathur, G., Shenoy, P.: Rethinking data management for storage-centric sensor networks. In: Proc. 3rd Biennial Conference on Innovative Data Systems Research (CIDR, Asilomar, CA, USA, January 2007) 22–32
6. Adler, M., Berenbrink, P., Friedetzky, T., Goldberg, L.A., Goldberg, P., Paterson, M.: A proportionate fair scheduling rule with good worst-case performance. In: Proc. 15th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA, San Diego, CA, USA, June 2003), New York, NY, USA, ACM Press (2003) 101–108
7. Bender, M.A., Fekete, S.P., Kröller, A., Liberatore, V., Mitchell, J.S.B., Polishchuk, V., Suomela, J.: The minimum-backlog problem. Presented at the 2nd International Conference on Mathematical Aspects of Computer and Information Sciences (MACIS, Paris, France, December 2007)
8. Dietz, P.F., Sleator, D.D.: Two algorithms for maintaining order in a list. In: Proc. 19th Annual ACM Symposium on Theory of Computing (STOC, New York, NY, USA, May 1987), New York, NY, USA, ACM Press (1987) 365–372
9. Few, L.: The shortest path and the shortest road through n points. *Mathematika* **2** (1955) 141–144