# Three Results on Frequency Assignment in Linear Cellular Networks

Marek Chrobak[*] Jiří Sgall[†]

**Abstract**

In the frequency assignment problem we are given a graph representing a wireless network and a sequence of requests, where each request is associated with a vertex. Each request has two more attributes: its arrival and departure times, and it is considered active from the time of arrival to the time of departure. We want to assign frequencies to all requests so that at each time step any two active requests associated with the same or adjacent vertices use different frequencies. The objective is to minimize the number of frequencies used.

We focus exclusively on the special case of the problem when the underlying graph is a linear network (path). For this case, we consider both the offline and online versions of the problem, and we present three results. First, in the incremental online case, where the requests arrive over time, but never depart, we give an algorithm with an optimal (asymptotic) competitive ratio $\frac{4}{3}$. Second, in the general online case, where the requests arrive and depart over time, we improve the current lower bound on the (asymptotic) competitive ratio to $\frac{11}{7}$. Third, we prove that the offline version of this problem is $\mathbb{NP}$-complete.

## 1 Introduction

**The frequency assignment problem.** In a wireless network, the coverage area is divided into cells, with each cell covered by a transmitter. Each user within a given cell is assigned a unique frequency for communicating with the transmitter. In order to avoid interferences, it is also necessary to ensure that any pair of adjoining cells uses different sets of frequencies. The set of available frequencies is a limited resource; thus the frequency assignment policy needs to attempt to minimize the total number of assigned frequencies.

In the *static* setting, where the set of users in each cell is fixed, we can model this problem as a variation of graph coloring, in which the network is represented by an undirected graph $G$ whose vertices represent the cells and edges connect adjacent cells. We are given an integer demand $w_v \geq 0$ (the number of users in cell $v$) for each vertex $v$. We wish to assign a set $C_v$ of $w_v$ colors to each vertex $v$, such that $C_u \cap C_v = \emptyset$ for all adjacent vertices $u, v$. The objective is to find such a color assignment that minimizes the total number of colors used.

It is natural to consider graphs $G$ with a regular structure. For example, in the literature it is commonly assumed that the cells are regular hexagons in the plane, in which case $G$ is a triangular

[*]Department of Computer Science, University of California, Riverside, CA 92521, USA. Research supported by NSF Grant CCF-0729071.

[†]Dept. of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Malostranské nám. 25, CZ-11800 Praha 1, Czech Republic. Email: `sgall@kam.mff.cuni.cz`. Partially supported by Inst. for Theor. Comp. Sci., Prague (project 1M0545 of MŠMT ČR) and grant IAA100190902 of GA AV ČR.

grid graph (see, for example, [6, 9, 2, 4], or the surveys in [8, 1]). As shown by McDiarmid and Reed [6] the static frequency assignment problem for such graphs is $\mathbb{NP}$-hard. They also gave a polynomial-time $\frac{4}{3}$-approximation algorithm for this version. Another $\frac{4}{3}$-approximation algorithm was designed independently by Narayanan and Shende [9].

**Dynamic setting.** Of course, in practice, the set of users in each cell is dynamic, as the users arrive and leave over time. Thus it is natural to study the frequency assignment in the *dynamic* setting. Here, each frequency request (user) has three attributes: a vertex $v$ where the request is issued, the arrival time and the departure time. The request is *active* between its arrival and departure times. We need to assign frequencies to all requests in such a way that at each time step, frequencies assigned to active requests at the same or adjacent vertices are different. The objective is to minimize the total number of frequencies used.

**Online algorithms.** For the dynamic setting described above, online algorithms are of particular significance, since in realistic applications users' arrivals and departures are unknown and unpredictable. In the online scenario, the arrival and departure events come over time, and the online algorithm has to react immediately; once a frequency is assigned to a request, it cannot be changed.

For an online frequency assignment algorithm $\mathcal{A}$, let $\mathcal{A}(I)$ be the number of frequencies used by $\mathcal{A}$ on an instance $I$, and let $opt(I)$ be the minimum number of frequencies required for $I$. We define $\mathcal{A}$ to be *R-competitive*, if there exists a constant $B$ independent of $I$ such that $\mathcal{A}(I) \leq R \cdot opt(I) + B$. When $B = 0$, we say that the ratio $R$ is *absolute*; otherwise, if we need to distinguish the cases, we say that the ratio $R$ is *asymptotic*. In this paper, we study the asymptotic competitive ratio.

In the *incremental* version of this problem, it is assumed that requests, once issued, last forever. This version corresponds to the static version of the offline problem, and it has been intensely studied. Chan *et al.* [2] derive some bounds on the asymptotic competitive ratio for hexagonal-cell graphs, proving that the optimal ratio is between 1.5 and 1.9126. For the absolute ratio, they give a tight bound of 2. More generally, for $\xi$-colorable graphs, they show an upper bound of $(\xi + 1)/2$.

Again, special classes of graphs can be studied. One natural and simple case, and the focus of our study, is that of a linear network (*path*), where the vertices are represented by integer points on the real line and two vertices are adjacent if they are at distance 1. For this case, it is known that the asymptotic competitive ratio is between $\frac{4}{3} \approx 1.333$ and $\frac{1}{2}(5 - \sqrt{5}) \approx 1.382$, while the optimal absolute ratio is equal to 1.5, see [3].

In the general version of the online problem departures are allowed, as in the dynamic version of the offline problem. For the problem on the path, Chan *et al.* [3] proved that the asymptotic competitive ratio is between $\frac{14}{9} \approx 1.556$ and $\frac{5}{3} \approx 1.667$, while the absolute competitive ratio is 1.5. Both upper bounds are achieved by a simple greedy strategy which always assigns the smallest frequency that can be used at that time.

**Our contribution.** We first consider the incremental case of the online problem on the path. We give an online algorithm with the asymptotic competitive ratio equal $\frac{4}{3}$, thus matching the lower bound and improving the upper bound of 1.382, both given in [3]. The idea of our algorithm is to spread the requests at each vertex evenly between four mutually overlapping sets of vertices, such that within each set we can allocate frequencies optimally with respect to this set. As each vertex belongs to three sets, the ratio $\frac{4}{3}$ will follow. The algorithm is very simple and 1-local, in the sense that the choice of each allocated frequency depends only on the frequencies used currently at the given vertex and its neighbors [4].

Next, we consider the general online case, where departures are allowed. For this case we prove a lower bound of $\frac{11}{7} \approx 1.571$, improving the currently best bound of $\frac{14}{9} \approx 1.556$ in [3]. Table 1 summarizes the known bounds for online algorithms for the path.

| Case | incremental | | with departures | |
|---|---|---|---|---|
| Ratio | absolute | asymptotic | absolute | asymptotic |
| lower bound | 1.5 | $4/3 \approx 1.333$ | $5/3 \approx 1.667$ | $\mathbf{11/7 \approx 1.571}$ |
| upper bound | 1.5 | $\mathbf{4/3 \approx 1.333}$ | $5/3 \approx 1.667$ | $5/3 \approx 1.667$ |

Table 1: Competitive ratios for the case of a path. Bounds from this paper are in boldface; all other bounds are from [3].

In the incremental case (on the path), it is easy to determine the value of the optimum: It is simply equal to the maximum number of frequencies on two adjacent vertices. This turns out to be false when departures are allowed. Our third result is that computing this optimum is $\mathbb{NP}$-hard. This proof uses a reduction from 3-coloring of planar graphs, and it shows that even deciding if three frequencies are sufficient is hard. Thus also achieving a better absolute approximation ratio than $\frac{4}{3}$ is $\mathbb{NP}$-hard. This result complements the $\mathbb{NP}$-hardness result in [6] and the offline upper bounds in [6, 9]. It also indicates that establishing the optimal competitive ratio for this case is likely to be more challenging, since it implies that there is probably no simple way to keep track of the optimum solution.

**Broader context.** The linear topology is of course of only limited practical significance. In fact, even the common model of hexagonal cells is only a convenient approximation; in reality the cells have rather complex geometrical shapes (see [7], for example). Nevertheless, further progress on more complex topologies is not likely without a complete understanding of frequency allocation in linear networks.

It is also worth pointing out at this point that the frequency allocation model we study is only one of many; in fact the frequency allocation problem in various scenarios has been studied since 1960's. We refer the reader to the surveys by Murthy *et al.* [8] and Aardal *et al.* [1] and book by Mishra [7] for information on other variants of frequency allocation.

## 2 Preliminaries

We identify vertices of the path with integers, $v = \ldots, -2, -1, 0, 1, \ldots$. Frequencies are denoted by positive integers. At each step, a request can be issued at some vertex $v$. We then need to assign a frequency to this request that is different from all frequencies already assigned to the active requests at vertices $v-1$, $v$ and $v+1$. The objective is to minimize the number of frequencies used at the same time.

For each vertex $v$, let $L_v$ be the (dynamic) set of frequencies assigned to $v$ by the algorithm and $\ell_v = |L_v|$. Thus $\ell_v$ is simply the number of active requests at $v$ at a given time. A frequency $f$ is called *admissible* for $v$ if $f \notin L_{v-1} \cup L_v \cup L_{v+1}$.

To estimate the performance of an algorithm, we measure the number of frequencies used. This is equivalent to measuring the maximum used frequency, as is done in some literature [3, 4, 4]. Clearly, the number of frequencies cannot exceed the maximum frequency. On the other hand, any algorithm can be modified to use only consecutive integers. This is trivial in the incremental case, as we can simply renumber the frequencies in the order in which they appear. Similarly, in the general case, if the online algorithm is about to use a frequency $f$ not used by any active request at this time, we change it to use the lowest unused frequency $f'$, and fix the mapping $f \mapsto f'$ for as long as there are some active requests with frequency $f$.

**The optimum.** As observed in [3], in the static case the optimum number of frequencies is

$$\omega \quad = \quad \max_v \{\ell_v + \ell_{v+1}\}. \tag{1}$$

Indeed, the "$\geq$" bound is trivial. To see that the "$\leq$" bound holds, we can assign frequencies to vertices as follows: If $v$ is even, assign to it frequencies $1, 2, \ldots, \ell_v$, and if $v$ is odd, assign to it frequencies $\omega, \omega - 1, \ldots, \omega - \ell_v + 1$. Then no two adjacent vertices will be assigned the same frequency.

Clearly, the optimum number of frequencies in the online incremental case is the same as the optimum for the static instance consisting of all requests from the online instance. Thus the formula (1) applies to the online incremental case as well, where $\ell_v$ is understood to be the number of all requests to $v$.

## 3 An Upper Bound for the Incremental Online Case

**Algorithm** FourBuckets. We partition all available frequencies $1, 2, 3, \ldots$ into four disjoint infinite *buckets* denoted $B_\sigma$, for $\sigma = 0, 1, 2, 3$. The frequencies in bucket $B_\sigma$ are denoted $1^\sigma, 2^\sigma, \ldots$, and are assumed to be ordered in this way, with $1^\sigma$ being the lowest one. How the partition into buckets is defined is not important. For example, one such partition can be achieved by defining $x^\sigma = \sigma + 4x - 3$, for each integer $x \geq 1$. For any vertex $v$ and $\sigma \in \{0, 1, 2, 3\}$, we say that $\sigma$ is *associated* with $v$ if $\sigma \not\equiv v \pmod 4$. Thus each vertex has three out of four buckets associated with it.

Suppose that a request is issued at a vertex $v$. Choose any $\sigma \in \{0, 1, 2, 3\}$ associated with $v$ that minimizes $|L_v \cap B_\sigma|$, and assign to this request the lowest frequency $f^\sigma$ from $B_\sigma$ admissible at $v$.

**Analysis.** The general idea is this: By the assignment of buckets to vertices, each bucket is associated with groups of exactly three consecutive vertices on the path. In each bucket, the algorithm is equivalent to the greedy algorithm, which is optimal for paths of three vertices. At each vertex, the algorithm spreads the requests evenly among the three buckets associated with this vertex, so each bucket gets about one third of all requests at each vertex. This implies that the number of frequencies used by the algorithm in each bucket is about $\frac{1}{3}$ of the optimum. Multiplying by the number of buckets, we conclude that the competitive ratio is $\frac{4}{3}$.

Now we give a formal argument. Let $L_{\sigma,v} = |L_v \cap B_\sigma|$ be the set of frequencies in $B_\sigma$ assigned to requests at a vertex $v$, and $\ell_{\sigma,v} = |L_{\sigma,v}|$ be the cardinality of $L_{\sigma,v}$. For any subset $X \subseteq B_\sigma$, we use notation $\max^\sigma(X)$ for the maximum integer $h$ such that $h^\sigma \in X$; for the empty set we put

$\max^\sigma(\emptyset) = 0$. The following lemma subsumes the proof that the greedy algorithm is optimal for paths of three vertices.

**Lemma 1.** *For each $\sigma$ and $v$, $\max^\sigma(L_{\sigma,v}) \leq \ell_{\sigma,v} + \max\{\ell_{\sigma,v-1}, \ell_{\sigma,v+1}\}$.*

*Proof.* Without loss of generality, we can assume that $\sigma = 0$ and $v \in \{0, 1, 2, 3\}$.

For $v = 0$, since $\sigma$ is not associated with $v$, we have $L_{0,0} = \emptyset$, so $\max^0(L_{0,0}) = 0$, and the lemma holds trivially.

Suppose $v = 1$, and let $h = \max^0(L_{0,1})$. By the algorithm, each frequency $1^0, \ldots, h^0$ is either in $L_{0,1}$ or $L_{0,2}$, as otherwise the algorithm would not use $h^0$. Of course, no frequency is in both sets. So $h \leq \ell_{0,1} + \ell_{0,2}$.

The case $v = 3$ is symmetric to the previous one.

Finally, consider the case $v = 2$, and let $h = \max^0(L_{0,2})$. Let $g = \max^0(L_{0,1} \cup L_{0,3})$; by symmetry, we can assume $g = \max^0(L_{0,1})$. Like in the previous case, each frequency $1^0, \ldots, g^0$ is in either $L_{0,1}$ or $L_{0,2}$, but not in both. Thus, for $h \leq g$, this implies that $h \leq \ell_{0,1} + \ell_{0,2}$. So assume $h > g$. When $h^0$ is assigned, each frequency $g^0 + 1, \ldots, h^0$ is in $L_{0,1} \cup L_{0,2} \cup L_{0,3}$, as otherwise we would not use $h^0$. However, by the definition of $g$, none of these frequencies is in $L_{0,1} \cup L_{0,3}$, thus all of them are in $L_{0,2}$. Therefore $h \leq \ell_{0,1} + \ell_{0,2}$ again. □

**Theorem 2.** *Algorithm* FourBuckets *is asymptotically $\frac{4}{3}$-competitive for the incremental frequency assignment on a path.*

*Proof.* Consider any vertex $v$ and any $\sigma$. By the algorithm, if $\sigma$ is not associated with $v$ then $\ell_{\sigma,v} = 0$. On the other hand, if $\sigma$ and $\sigma'$ are associated with $v$ then $\ell_{\sigma,v} \leq \ell_{\sigma',v} + 1$. This implies that for each $\sigma$ associated with $v$ we have $\ell_{\sigma,v} \leq \lceil \ell_v/3 \rceil \leq \frac{1}{3}(\ell_v + 2)$. Therefore, using Lemma 1, we get

$$\begin{aligned}
\max^\sigma(L_{\sigma,v}) &\leq \ell_{\sigma,v} + \max\{\ell_{\sigma,v-1}, \ell_{\sigma,v+1}\} \\
&\leq \tfrac{1}{3}(\ell_v + 2) + \tfrac{1}{3}\max\{\ell_{v-1} + 2, \ell_{v+1} + 2\} \\
&\leq \tfrac{1}{3}[\ell_v + \max\{\ell_{v-1}, \ell_{v+1}\} + 4] \\
&\leq \tfrac{1}{3}(\omega + 4).
\end{aligned}$$

Thus $|B_\sigma| = \max_v \max^\sigma(L_{\sigma,v}) \leq \frac{1}{3}(\omega + 4)$ as well, and we can conclude that the total number of frequencies used in all four buckets is at most $\frac{4}{3}(\omega + 4)$. □

The additive constant in the analysis above has not been optimized and it can be improved by considering cases depending on the residue of $\ell_v$ modulo 4. We leave this as an exercise for the reader.

## 4 A Lower Bound for the General Online Case

To obtain an improved lower bound, we modify the idea from [3, Section 4.2], which we first describe informally.

Suppose that we start with $k$ requests on vertices 0 and 2 each. Next, if the total number of frequencies does not exceed $\frac{3}{2}k$, we can remove $\frac{1}{2}k$ appropriate requests from each of these vertices so that the online algorithm uses $k$ distinct frequencies in total for the remaining requests. Then we issue $\frac{1}{2}k$ requests on vertex 1. The online algorithm will use $\frac{3}{2}k$ frequencies, while the optimum is $k$. This gives us a lower bound of $\frac{3}{2}$.

The first observation, explored in [3], is that if initially the two $k$-tuples of requests use slightly distinct sets of frequencies, then the idea above can be refined to give a better lower bound; this is described below in Procedure FinishOff.

Furthermore, if we start with two $k$-tuples of requests on vertices 0 and 2, we can use the key ingredient from [3] (see Procedure Expand below) to create two $k$-tuples of requests at distance 2 apart that are served by two sets of frequencies that differ more than the initial two sets. To this end, issue $k$ requests on vertex 5. If many new frequencies are used, we are done. Otherwise, we can remove some requests from 0 and 2 and then add $k$ requests at 1 so that they use many frequencies distinct from these at 5. Now, remove all requests from 0 and 2. The last trick is to issue $k$ requests at 3, and these must use at least half as many distinct frequencies either from the requests at 1 or from the requests at 5. Remove all requests either from 5 or 1, whichever vertex has more frequencies in common with vertex 3. As it turns out, we end up with two vertices, each having $k$ active request, whose frequency sets differ by more than the two initial frequency sets on vertices 0 and 2.

In [3], the $\frac{14}{9} \approx 1.556$ lower bound is obtained by running first Procedure Expand followed by Procedure FinishOff. We improve the bound by iterating Procedure FinishOff. A somewhat careful argument is needed to show that the overall optimum is still $k$. Optimizing the parameters, we get the lower bound of $\frac{11}{7} \approx 1.571$.

**Theorem 3.** *No deterministic online algorithm for general online frequency assignment on a path with 8 vertices has competitive ratio smaller than* $\frac{11}{7} \approx 1.571$.

*Proof.* Let $R = \frac{11}{7}$ be our target competitive ratio. Let $\rho > 0$ be small and $k$ be a sufficiently large positive integer. We give an adversary strategy which, for a given online algorithm $\mathcal{A}$, generates a sequence on which $\mathcal{A}$ uses at least $(R-\rho)k$ frequencies while the optimum is $k$. By taking $\rho$ small and $k$ large, we obtain the desired lower bound.

We first describe the overall adversary strategy and the two procedures FinishOff and Expand. Then we verify that the optimum is $k$.

At the beginning, the adversary simply issues $k$ requests at vertex 0 and $k$ requests at vertex 2. The rest of the adversary strategy is divided into phases. The invariant at the beginning of each phase is that there are two vertices $v$ and $v+2$ with $|L_v| = |L_{v+2}| = k$, and that there are no other active requests. In each phase, the adversary proceeds as follows. Let $s$ and $\delta$ be such that the online algorithm now uses $|L_v \cup L_{v+2}| = s = (1+\delta)k$ frequencies. If $\delta \geq \frac{1}{7} - \rho$, then the adversary completes the sequence by executing Procedure FinishOff. Otherwise, it uses Procedure Expand; this either completes a sequence or ends in a configuration with $k$ requests on each of two vertices $u$ and $u+2$, for some $u$, in which case we continue with the next phase.

Procedure FinishOff: Let $U$ be the $\lceil k - \frac{1}{2}s \rceil$ lowest frequencies in $L_v \cap L_{v+2}$ and let $U'$ be the $\lceil k - \frac{1}{2}s \rceil$ highest frequencies in $L_v \cap L_{v+2}$. The adversary removes the requests using frequencies $U$ from $v$ and the requests using frequencies $U'$ from $v+2$. Next, he makes $\lceil k - \frac{1}{2}s \rceil$ requests on $v+1$. These requests will have to be assigned frequencies other than those left at $v$ and $v+2$. The remaining frequencies at $v$ and $v+2$ are distinct, with $\lfloor \frac{1}{2}s \rfloor$ frequencies at each vertex. Thus the current total number of frequencies used is at least $k + \lfloor \frac{1}{2}s \rfloor \geq \frac{1}{2}(3+\delta)k - 1$. Note that if $\delta \geq \frac{1}{7} - \rho$ then the number of frequencies is at least $(\frac{11}{7} - \frac{1}{2}\rho)k - 1 \geq (R-\rho)k$, for a sufficiently large $k$.

Procedure Expand: Since the path has eight vertices, it must contain either vertex $v+5$ or $v-3$. Without loss of generality we suppose that it contains $v+5$; the other case is symmetric.

Issue $k$ requests on $v+5$. If $|L_v \cup L_{v+2} \cup L_{v+5}| \geq Rk$, we stop the input sequence. In the remaining case we have $|L_v \cup L_{v+2} \cup L_{v+5}| \leq Rk$. This implies two things. First, denoting $r = (2+\delta-R)k$, we get

$$\begin{aligned} |(L_v \cup L_{v+2}) \cap L_{v+5}| &= |(L_v \cup L_{v+2})| + |L_{v+5}| - |L_v \cup L_{v+2} \cup L_{v+5}| \\ &\geq (1+\delta)k + k - Rk \;=\; r. \end{aligned}$$

Second, for each $u \in \{v, v+2\}$, we obtain

$$\begin{aligned} |L_u \cap L_{v+5}| &= |L_u| + |L_{v+5}| - |L_u \cup L_{v+5}| \\ &\geq |L_u| + |L_{v+5}| - |L_v \cup L_{v+2} \cup L_{v+5}| \\ &\geq (2-R)k \;\geq\; \tfrac{1}{2}r, \end{aligned}$$

where the last inequality uses the fact that $\delta < \frac{1}{7}$ whenever we use Expand. Therefore there are sets $U \subseteq L_v \cap L_{v+5}$ and $U' \subseteq L_{v+2} \cap L_{v+5}$ such that $U \cap U' = \emptyset$ and $|U| = |U'| = \lfloor \frac{1}{2}r \rfloor$. Remove from $v$ all the requests that do not use frequencies in $U$ and from $v+2$ all the requests that do not use frequencies in $U'$. Then issue $k - \lfloor \frac{1}{2}r \rfloor$ requests on $v+1$. These requests will have to be allocated frequencies that are not in $U \cup U'$, while those in $U \cup U' \subseteq L_{v+5}$ are still used at $v+5$. Thus at this point we have $|L_{v+1} - L_{v+5}| \geq \lfloor \frac{1}{2}r \rfloor$. Then delete all the remaining requests from $v$ and $v+2$ and issue $\lfloor \frac{1}{2}r \rfloor$ more requests on $v+1$. As a result, we have $|L_{v+1}| = |L_{v+5}| = k$ and $|L_{v+1} \cup L_{v+5}| = k + z$, for $z \geq \lfloor \frac{1}{2}r \rfloor$.

Next, the adversary makes $k$ requests on $v+3$. Then we have $\max\{|L_{v+1} \cup L_{v+3}|, |L_{v+3} \cup L_{v+5}|\} \geq k + \frac{1}{2}z$, so, without loss of generality, we can assume that $|L_{v+1} \cup L_{v+3}| \geq k + \frac{1}{2}z$. Finally, the adversary removes all requests from $L_{v+5}$. Note that at this time we have $|L_{v+1}| = |L_{v+3}| = k$, $|L_{v+1} \cup L_{v+3}| \geq k + \frac{1}{2}z \geq (\frac{3}{2} + \frac{1}{4}\delta - \frac{1}{4}R)k - 1$, and that no vertex other than $v+1$, $v+3$ has any active requests. This completes the description of Procedure Expand.

To finish the description of the adversary strategy, we need to show that it will terminate after finitely many phases. Suppose that we have a phase that is not a final one. Thus this phase starts with $\delta = \frac{1}{7} - \rho - \epsilon$ for some $\epsilon > 0$ and uses Expand. At the end, the number of used frequencies is at least

$$\begin{aligned} (\tfrac{3}{2} + \tfrac{1}{4}(\tfrac{1}{7} - \rho - \epsilon) - \tfrac{1}{4}R)k - 1 &= (1 + \tfrac{1}{7} - \tfrac{1}{4}(\rho + \epsilon))k - 1 \\ &\geq (1+\delta')k, \end{aligned}$$

for $\delta' = \delta + \frac{1}{2}\rho$ and sufficiently large $k$ (independent of $\epsilon$). Thus, for any $\rho > 0$ and a sufficiently large $k$, after a fixed number of phases, $\delta$ increases above $\frac{1}{7} - \rho$, at which point the adversary uses FinishOff and completes the sequence.

The description of the strategy shows that at the end algorithm $\mathcal{A}$ uses at least $(R-\rho)k$ frequencies.

To finish the proof, it remains to show that the optimum number of frequencies is $k$. Given the instance produced by the adversary strategy, we will maintain an offline solution (that is, a dynamic frequency assignment). In addition to using only given $k$ frequencies, this offline frequency assignment will maintain the following invariant:

($*$) Let $U$ and $U'$ be the sets of frequencies (used by $\mathcal{A}$) from the description of the procedure FinishOff or Expand in the current phase. Consider the corresponding sets of requests, i.e., the requests at $v$ using frequencies $U$ and the requests at $v+2$ using frequencies $U'$. Then these two sets of requests use the same set of frequencies.

At the beginning of the first phase, after the first $2k$ requests, we can guarantee the invariant while using only $k$ frequencies total, since we can assign the frequencies arbitrarily. Next, we check that for both FinishOff and Expand we can serve the sequence with $k$ frequencies and maintain the invariant.

In FinishOff, after removing the requests corresponding to $U$ and $U'$, the same frequencies are used at $v$ and $v+2$, by invariant ($*$). So we have enough admissible frequencies for the new requests at $v+1$ (among the $k$ original frequencies). This is the last phase, so there is no invariant to be maintained.

In Expand, first we assign the requests at $v+5$ the same $k$ frequencies as at $v$ and $v+2$. If the sequence does not stop now, then, by invariant ($*$), after removing the requests not corresponding to $U$ and $U'$, the same frequencies are used at $v$ and $v+2$. So we have enough admissible frequencies for the first batch of requests at $v+1$. Next we remove the remaining requests at $v$ and $v+2$, thus we have admissible frequencies for the remaining requests at $v+1$. Finally, for the $k$ requests at $v+3$, we may assign the $k$ frequencies arbitrarily. In particular, we can guarantee the invariant for the next phase.

Thus the optimum uses only $k$ frequencies and the competitive ratio is at least $R = \frac{11}{7}$, completing the proof of the lower bound. □

We remark that the value $R = \frac{11}{7}$, as well as the choice of the breakpoint $\delta = \frac{1}{7}$, is optimal for the strategy described in the proof.

## 5 $\mathbb{NP}$-Completeness

As we have seen, computing the optimum in the incremental version is easy. Now we show that this is not the case once we allow dynamic requests.

Let FAPP stand for the decision version of the frequency allocation problem for the path: "Given a sequence of requests and an integer $k$, determine whether these requests can be served with at most $k$ frequencies".

**Theorem 4.** *The problem* FAPP *is $\mathbb{NP}$-complete, even for any fixed $k \geq 3$.*

*Proof.* We reduce the 3-coloring problem for planar graphs (3CPG) to FAPP. Suppose that $G$ is a planar graph. We construct a sequence $I$ of requests such that $I$ can be served with 3 frequencies if and only if $G$ has a 3-coloring.

Using the result of [5], we can embed $G$ in a plane so that (i) each vertex is represented by a vertical line segment, and (ii) each edge $(u, v)$ is represented by a horizontal line segment whose endpoints connect the vertical segments representing $u$ and $v$, without intersecting any other vertical segments. (See Figure 1.) This embedding can also be obtained from visibility representations of planar graphs (see, for example, [10, 11]). We align all segments so that coordinates of all their endpoints are integral and multiples of 6.

Now we construct an instance $I$ of FAPP. The x-axis of the plane will correspond to the path and the y-axis will represent time. It is convenient to think of requests in $I$ as vertical intervals in

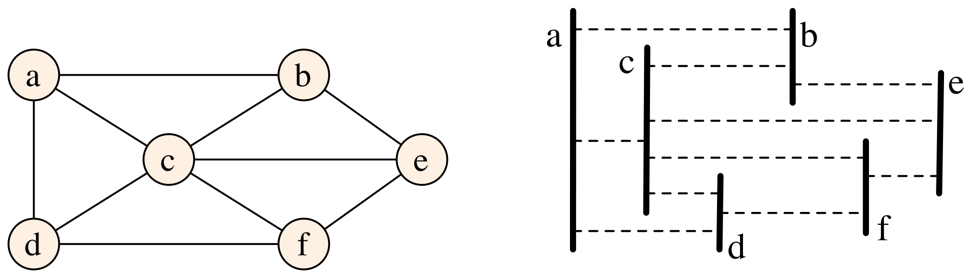

Figure 1: A graph and its embedding.

the plane. We will denote each such interval by $(x, y', y'')$, where $x'$ is its x-coordinate and $y', y''$ the bottom and top y-coordinates. As a request, it is a request at vertex $x$ (of the path) arriving at time $y'$ and departing at time $y''$.

For each vertex in $G$ represented by a vertical segment $(x, y', y'')$, the instance contains the request $(x, y', y'')$. For each edge represented by a segment from $(x', y)$ to $(x'', y)$, where $x'' > x'$, we build a gadget consisting of a number of requests. Note that $x'' - x'$ is even and at least 6, by our assumptions. We add two requests $(x'+1, y+4, y+5)$, one request $(x'+2, y+2, y+5)$, and two requests $(x'+2, y, y+3)$. Furthermore we will add requests $(x'+i, y, y+1)$ for any $i$, $2 < i < x''-x'$; we add one such request if $i$ is odd and two such requests if $i$ is even. By the construction and alignment of the segments, the requests from different edge gadgets do not interfere. (See Figure 2.)

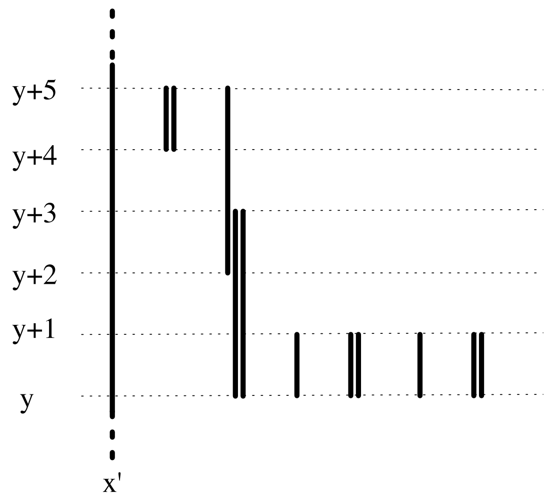

Figure 2: An edge gadget.

The edge gadget guarantees that the assignment of frequencies to (the requests corresponding to) its endpoints can be extended to an assignment of frequencies to the request in the edge gadget using only three frequencies in total if and only if the two vertex colors are distinct. Consider again an edge represented by a segment from $(x', y)$ to $(x'', y)$, for $x'' > x'$. Suppose that the request containing point $(x', y)$ (i.e., the first endpoint of an edge) is assigned frequency $f$. We show that if only three frequencies $f, f', f''$ are used, the assignment is essentially unique. The two requests $(x'+1, y+4, y+5)$ must use $f'$ and $f''$. Now the request $(x'+2, y+3, y+5)$ must use $f$, and the two requests $(x'+2, y, y+3)$ must use $f'$ and $f''$. We continue by induction to show that for $i$ odd, $(x'+i, y, y+1)$ uses $f$ and, for $i$ even, the two requests $(x'+i, y, y+1)$ use $f'$ and $f''$. This implies that $(x''-1, y, y+1)$ uses $f$. Overall, the assignment is valid if and only if the request containing $(x'', y)$ is not colored by $f$.

This implies that $G$ is 3-colorable if and only if the instance $I$ of FAPP has a frequency allocation with only 3 frequencies.

The extension to $k \geq 4$ is easy. We use the same construction as for $k = 3$, with some additional requests. We will produce an instance $I'$ of FAPP that contains all requests from the previously described instance $I$. To describe the new requests, let $\bar{x}$ and $\bar{y}$ be, respectively, the largest x- and y-coordinate used in $I$ (with the minimum coordinates assumed to be 0). For all $x = 0, \ldots, \bar{x}$, if $x$ is even, we add three requests $(x, \bar{y}+1, \bar{y}+2)$, and if $x$ is odd we add $k-3$ requests $(x, 0, \bar{y}+2)$. As a result, in every allocation of $k$ frequencies to $I'$, all requests $(x, 0, \bar{y}+2)$ will use the same set of $k-3$ frequencies, say $4, 5, \ldots, k$. Consequently, the requests in $I$ will be assigned frequencies $1, 2, 3$. By the same argument as before, we obtain that $G$ is 3-colorable if and only if $I'$ has a frequency allocation with only $k$ frequencies. $\square$

We point out that Theorem 4 is not comparable to the $\mathbb{NP}$-hardness result in [6], as our result applies to the dynamic case and linear networks, while the result in [6] applies to the static case for hexagonal cells. (The reductions are significantly different as well.)

## 6 Final Comments

The most outstanding open problem is to establish the optimal asymptotic competitive ratio for the dynamic case (with expirations) on the path. The current gap is between $\frac{11}{7}$ and $\frac{5}{3}$.

We would like to point out that the idea of Algorithm FourBuckets can be generalized as follows. Suppose that $G$ has $p$ induced subgraphs $G_1, \ldots, G_p$ with the following properties: (1) Each vertex of $G$ belongs to exactly $q$ subgraphs $G_i$, (2) Each subgraph $G_i$ does not contain a 4-path (that is, each $G_i$ is a collection of disjoint stars). Then $G$ has a $p/q$-competitive algorithm. In fact, this can be generalized further: if, instead of (2), we require that all $G_i$ have an $R$-competitive algorithm, then $G$ will have a $pR/q$-competitive algorithm. As of now, however, we have not been able to apply it to improve upper bounds for other types of graphs.

## Acknowledgments.

We would like to thank Deshi Ye and Guochuan Zhang for introducing us to the frequency allocation problem and stimulating discussions. We are also grateful to Dan Král and Jan Kratochvíl for discussions on graph drawing and to the anonymous referees for comments that helped us improve the presentation.

## References

[1] K. Aardal, S. P. M. van Hoesel, A. M. C. A. Koster, C. Mannino, and A. Sassano. Models and solution techniques for frequency assignment problems. *4OR: Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 1(4):261–317, 2003.

[2] J. W.-T. Chan, F. Y. L. Chin, D. Ye, and Y. Zhang. Online frequency allocation in cellular networks. In *Proc. 19th Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 241–249, 2007.

[3] J. W.-T. Chan, F. Y. L. Chin, D. Ye, Y. Zhang, and H. Zhu. Frequency allocation problem for linear cellular networks. In *Proc. 17th International Symp. on Algorithms and Computation (ISAAC)*, pages 61–70, 2006.

[4] F. Y. L. Chin, Y. Zhang, and H. Zhu. A 1-local 13/9-competitive algorithm for multicoloring hexagonal graphs. In *Proc. 13th Annual International Computing and Combinatorics Conf. (COCOON)*, pages 526–536, 2007.

[5] P. Duchet, Y. O. Hamidoune, M. Las Vergnas, and H. Meyniel. Representing a planar graph by vertical lines joining different levels. *Discrete Mathematics*, 46:319–321, 1983.

[6] C. McDiarmid and B. Reed. Channel assignment and weighted colouring. *Networks*, 36:114–117, 2000.

[7] A. R. Mishra. *Fundamentals of Cellular Network Planning and Optimisation: 2G/2.5G/3G... Evolution to 4G.* John Wiley & Sons, 2004.

[8] R. A. Murphey, P. M. Pardalos, Mauricio, and M. G. Resende. Frequency assignment problems. In *Handbook of Combinatorial Optimization*, pages 295–377. Kluwer Academic Publishers, 1999.

[9] L. Narayanan and S. M. Shende. Static frequency assignment in cellular networks. *Algorithmica*, 29(3):396–409, 2001.

[10] P. Rosenstiehl and R. Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Discrete & Computational Geometry*, 1:343–344, 1986.

[11] R. Tamassia and I. G. Tollis. A unified approach to visibility representation of planar graphs. *Discrete & Computational Geometry*, 1:321–341, 1986.