University of Patras - School of Engineering
Department of Computer Engineering
and Informatics

Independent Sets and Graph Coloring with Applications to the Frequency Allocation Problem in Wireless Networks

PhD Thesis

Evi Papaioannou

Advisor: Christos Kaklamanis, Professor

Patras, November 2004

Independent Sets and Graph Coloring with Applications to the Frequency Allocation Problem in Wireless Networks

PhD Thesis

EVI PAPAIOANNOU

Examination Committe:

Christos Kaklamanis, Professor (Advisor)
Lefteris Kirousis, Professor
Paul Spirakis, Professor
Athanasios Tsakalidis, Professor
Stavros Cosmadakis, Professor
Christos Zaroliagis, Associate Professor
Sotiris Nikoletseas, Assistant Professor

Independent Sets and Graph Coloring with Applications to the Frequency Allocation Problem in Wireless Networks

Evi Papaioannou

Abstract

The subject of this dissertation thesis is the study of issues arising in communication networks that utilize Frequency Division Multiplexing (FDM). We consider networks based on telecommunication infrastructure such as cellular mobile telephone networks and networks of autonomous transmitters, like ad hoc wireless networks. We represent these networks using graphs and model the corresponding communication problems as combinatorial optimization problems in such graphs. Our results include new on-line algorithms which outperform previously known algorithms as well as new lower bounds.

In cellular networks, a geographical region is virtually divided into subregions called cells. Each cell is the calling area of a base station which enables wireless communication. Each base station is responsible for servicing users located within its range. Communication between two users of a cellular network involves in the first place communication between each user and the base station that services the cell where the user is located. Then, communication between the base stations must be established. Wireless communication between users and their base station is always involved, even when both of the users are located in the same cell or only one of them uses the cellular network and the other uses, for example, the standard telephone network. Two users located close to each other can simultaneously communicate with their base station via Frequency Division Multiplexing technology, using different frequencies. Usually, the use of the same frequency by users located in the same or adjacent cells causes signal interference, thus, making communication hard or impossible to establish. The basic cellular network model assumes base stations equipped with equivalent transmitters, uniformly distributed on the plane. Therefore, the plane is divided in cells of hexagon shape. In fact, cells can have irregular shape and it may be the case that the signal interference constraints regarding the reuse of frequencies by different users are even harder and more complicated. We

model such constraints through interference graphs where nodes correspond to cells of the network and an edge between two nodes u and v states that the assignment of the same frequency to users located in cells corresponding to the nodes u and v would cause signal interference. In cellular networks, an important communication problem to be solved is the assignment of frequencies to the users so that they can communicate with their base station without signal interference. Since the available frequency spectrum is limited, its efficient utilization is critical and essential. Frequency allocation and call control are the most fundamental problems to be solved for the efficient use of the frequency spectrum and can be defined as follows:

- The frequency allocation problem is to assign frequencies to the users so that signal interference is avoided, while minimizing the total number of frequencies used.
- The call control problem in a network that supports a spectrum of w available frequencies is to assign frequencies to users so that signal interference is avoided, while maximizing the number of users served.

We consider the on-line version of both the frequency allocation and the call control problems. We follow the competitive analysis approach and we use the competitive ratio as a measure for evaluating the performance of an algorithm. Calls are not known a priori and appear gradually. When a new call appears, an on-line frequency allocation algorithm must assign it a frequency so that signal interference with calls previously appeared is avoided. An on-line call control algorithm must either reject (not assign a frequency) or accept the call by assigning it one of the w available frequencies so that signal interference is avoided. In both cases, the decisions of the algorithm cannot change in the future. The competitive ratio of an on-line frequency allocation algorithm indicates the relation between the number of frequencies used by the algorithm and the number of frequencies used in an optimal assignment. The competitive ratio of an on-line call control algorithm indicates the relation between the number of calls accepted by the algorithm and the maximum number of calls that could have been accepted. For the frequency allocation problem, we present an almost tight analysis of the greedy algorithm. For the call control problem, we present a series of randomized on-line algorithms with competitive ratios that deterministic algorithms cannot achieve. Our algorithms are simple in the sense that they use a small number of random bits or weak random sources for making their random choices. Our results can be extended to cellular networks with various interference graphs, like cellular networks with high reuse distance, irregular networks with interference graphs of small degree, as well as networks with arbitrary interference graphs. We also present new corresponding lower bounds for randomized on-line call control algorithms.

The frequency allocation and the call control problems are also essential in the case of networks of autonomous transmitters. In such networks, transmitters are distributed in the plane, each of them covers a circular region whose range depends on the technical specification of the transmitter (e.g. its transmission power) and can operate at a different frequency. We assume that transmission without signal interference requires that the range of each transmitter does not overlap with the range of other transmitters operating at the same frequency. Possible signal interference between transmitters operating at the same frequency is modelled by disk graphs, where each node corresponds to a circular region on the plane so that two such regions overlap or are adjacent if and only if there exists an edge connecting the corresponding nodes in the graph. If we consider frequencies as colors, the frequency allocation problem in networks of transmitters is equivalent to the minimum coloring problem in the disk graph modelling possible signal interference. The important special case of the call control problem where each transmitter operates at a particular frequency is equivalent to the maximum independent set problem in the corresponding disk graph.

We study the on-line version of both the maximum independent set and the minimum coloring problems in disk graphs. Again, we follow the competitive analysis approach and use the competitive ratio as a measure for the evaluation of the performance of our algorithms. The nodes of the graph are not known in advance and appear gradually. When a new node appears, an on-line coloring algorithm must assign it a color different from those assigned to previously appeared adjacent nodes. An on-line independent set algorithm starts from an empty set of nodes and gradually expands it. When a new node appears, the algorithm must decide whether to include this node in the set. In both cases, the on-line algorithm cannot change its decisions in the future. The competitive ratio of an on-line coloring algorithm indicates the relation between the number of colors used by the algorithm and the number of colors in an optimal coloring. The competitive ratio of an on-line independent set algorithm indicates the relation between the size of the independent set computed by the algorithm and the size of the maximum independent set.

For the independent set problem in disk graphs, we present the first new randomized on-line algorithms. We prove upper and lower bounds for their competitive ratios either if the disk representation is given as part of the input or if only the nodes and edges of the disk graph constitute the input. Obviously, the latter case is harder. When the disk representation is given as part of the input, we present on-line algorithms with significantly better competitive ratios than the known results in the literature obtained by deterministic on-line algorithms. Note that, the competitive ratios achieved by our algorithms outperform those obtained by deterministic on-line algorithms. Furthermore, our algorithms can extend also to more general cases that model the call control problem in networks of transmitters obtaining similar competitive ratios. When the disk representation is not given as part of the input, we prove lower bounds indicating that a particularly simple, intuitive, deterministic on-line algorithm can achieve a competitive ratio which is asymptotically optimal. This particular result implies that the use of randomization cannot help in improving the competitiveness of on-line independent set algorithms. For the coloring problem in disk graphs, we present a new on-line algorithm that does not make use of the disk representation and achieves a competitive ratio similar to the one obtained by the best-known algorithm in the literature that uses the disk representation. This particular result significantly improves the best previously known upper bound for on-line coloring algorithms that do not use the disk representation.

Contents

Chapter 1

Introduction

In the area of mobile communications, which combines wireless and high speed networking technologies, rapid technological progress has been made during the recent years through the development and use of flexible wireless and mobile networks (e.g., mobile telephone networks, bluetooth, wireless ad hoc networks), sensor networks, wireless broadband networks, etc. It is expected that in the near future, mobile users have access to a wide variety of services available over mobile communication networks. Nevertheless, there is still a lack in innovative methods and techniques that will allow exploitation of the new transmission media and guarantee efficient access to them. The use of the new communication media is based on techniques that have been successfully used in the past, but prove to be practically inefficient and insufficient nowadays. Therefore, the development of new protocols and algorithms that will allow for efficient use of the new communication media exploiting their special features is more than necessary.

The frequency allocation problem has many applications in the area of radio communication networks as well as of mobile cellular networks. It concerns the assignment of frequencies to transmitters so that signal interference (which appears when transmitters located in the same or neighboring geographical areas operate at the same or neighboring frequencies) is avoided or minimized. Due to the continuously increasing communication needs which tend to exceed the corresponding expansion of the available spectrum, the necessity for efficient management of this spectrum imposes the use of optimal or near optimal frequency allocation strategies. When a particular frequency is used in some cell of the network it cannot be used by another call in the same or adjacent cells, since in this case signal interference would be caused. On the other hand, each frequency can be used to service calls in geographically distant cells on the network (frequency reuse). In wireless communication networks transmitters have usually overlapping ranges. If we assume that a receiver uses a particular frequency, it is known that in the presence of signal interference due to other transmitters, the received signal is degraded. Interference can be caused by transmitters that either use the same frequency and are located in the same region, or use "neighboring " frequencies, since neither transmitters nor receivers operate exactly at a predefined frequency. The problem in this class of networks is , given a set of transmitters, to define the frequency at which each of them should operate, so that the signal interference, the maximum number of frequencies allocated and the number of different frequencies used are minimized. Efficient frequency allocation techniques in wireless networks must guarantee that signal interference is avoided and frequency reuse is maximized, so that the maximum number of frequencies is available in each cell and at the same time the number of the calls that the network can service is maximized. Such algorithms and techniques are categorized based on how they are executed and also on the way frequencies are divided in the network. In the case of fixed allocation algorithms, there is an initial assignment of frequencies to the cells so that their requirements are met (e.g. more frequencies are assigned to a cell covering a densely populated area) and the maximum possible reuse of frequencies is ensured. Fixed allocation algorithms are usually simple but not easily adaptive to workload (call) changes in the network. Borrowing techniques have been proposed for efficiently dealing with this problem (when there are increased requirements in an area with few available frequencies). Dynamic frequency allocation algorithms can more efficiently manage changes in the workload but have increased complexity and requirements in computational power. Another technique for designing and analyzing dynamic frequency allocation algorithms is the competitve analysis of on-line algorithms (algorithms that decide which requests to service without having knowledge of the requests that will appear in the future; this is what actually happens in real cellular networks). According to competitive analysis, the performance (in terms of the number of requests finally serviced) of an on-line algorithm for a sequence of calls not known a priori is compared to the performance of the optimal algorithm on the same sequence.

1.1 Frequency allocation and call control on wireless networks

We consider wireless networks in which base stations are used to build the required infrastructure. In such systems, the architectural approach used is the following. A geographical area in which communication takes place is divided into regions. Each region is the calling area of a base station. Base stations are connected via a high speed network.When a user A wishes to communicate with some other user B, a path must be established between the base stations of the regions in which the users A and B are located. Then communication is performed in three steps: (a) wireless communication between A and its base station, (b) communication between the base stations, and (c) wireless communication between B and its base station. Thus, the transmission of a message from A to B first takes place between A and its base station, the base station of A sends the message to the base station of B which will transmit it to B. At least one base station is involved in the communication even if both users are located in the same region or only one of the two users is part of the wireless network (and the other uses for example the PSTN).

Many users of the same region can communicate simultaneously with their base station. This can be achieved via frequency division multiplexing (FDM). The base station is responsible for allocating distinct frequencies from the available spectrum to users so that signal interference is avoided. Since the spectrum of available frequencies is limited, important engineering problems related to the efficient reuse of frequencies arise.

Signal interference in cellular networks can be represented by an *interference graph* G. Vertices of the graph correspond to cells and an edge (u, v) in the graph indicates that the assignment of the same frequency to two users lying at the cells corresponding to nodes u and v will cause signal interference. If the assumption of uniform distribution of identical base stations does not hold, arbitrary interference graphs can be used to model the underlying network. Such a network as well as the corresponding interference graph are depicted in Figure 1.1.

In the ideal case, base stations with equivalent transmitters are uniformly distributed within the network and the calling area of each base station is a circle which, for simplicity reasons, can be represented as a regular hexagon. Such networks are called ideal wireless cellular networks. An ideal wireless cellular network (and the corresponding interference graph) is depicted in the left part of Figure 1.2.

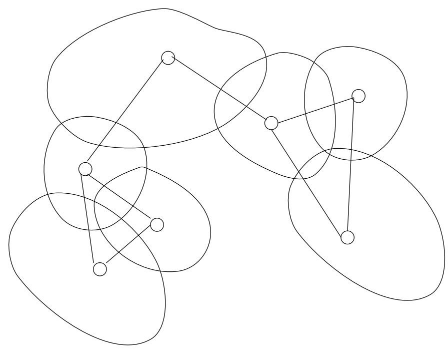

Figure 1.1: A wireless cellular network and the corresponding interference graph.

Signal interference usually manifests itself when the same frequency is assigned to users located in the same or adjacent cells. Alternatively, in this case, we may say that the cellular network has *reuse distance* 2. By generalizing this parameter, we obtain cellular networks of reuse distance k in which signal interference between users that have been assigned the same frequency is avoided only if the users are located in cells with distance at least k. The interference graph of a cellular network of reuse distance 2 is depicted in the right part of Figure 1.2.

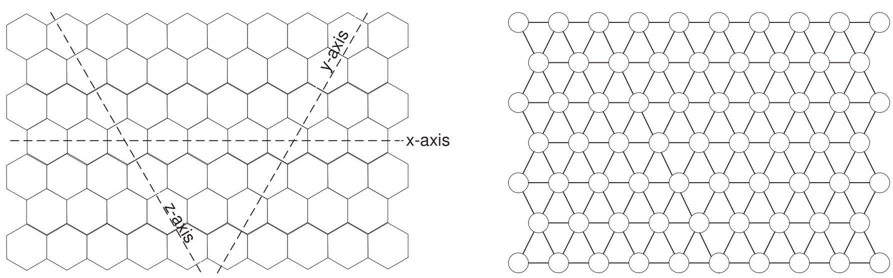

Figure 1.2: A cellular network and the corresponding interference graph when the reuse distance is 2.

We define as degree of a cell v the number of its neighboring cells. The network degree is the maximum degree over all cells of the network. Equivalently, the network degree is the maximum degree of the corresponding interference graph.

In wireless networks, given users that wish to communicate, two problems are of main interest and are addressed in this PhD Thesis:

- The *frequency allocation* problem is to assign frequencies to the users so that signal interference is avoided, minimizing the total number of frequencies used.

- The *call control* (or call admission) problem in a network that supports a spectrum of w available frequencies is to assign frequencies to users so that signal interference is avoided, maximizing the number of users served.

We assume that calls corresponding to users that wish to communicate appear in the cells of the network in an on–line manner. When a call arrives, an on–line frequency allocation algorithm accepts the call assigning a frequency to it, while a call–control algorithm decides either to accept the call (assigning a frequency to it), or to reject it. In both cases the objective is maximizing the benefit, i,e., using the fewer possible frequencies and servicing as many users as possible, respectively, while at the same time avoiding signal interference. Once a call is accepted, it cannot be rejected (preempted). Furthermore, the frequency assigned to the call cannot be changed in the future. We assume that all calls have infinite duration; this assumption is equivalent to considering calls of the same duration.

1.2 Independent sets and graph coloring

The frequency allocation problem in wireless networks is equivalent to the problem of multicoloring the nodes of a graph. In particular, given the interference graph of a wireless cellular network, if we imagine frequencies as colors, then allocating frequencies to users of the network is equivalent to multicoloring the nodes of the interference graph. The coloring must use the minimum number of colors (i,e., the communication between users must be obtained with the use of the minimum number of frequencies) and nodes of the interference graph connected with an edge must be assigned different colors (i,e., users located ““close”” to each other according to the reuse distance of the network, must be assigned different frequencies).

The call control problem in wireless networks is a generalization of the maximum independent set problem in graphs. Also in this case, given the interference graph of a cellular wireless network and a limited spectrum of w available frequencies, considering frequencies as colors, what we have to achieve is to color as many nodes of the graph as we (i,e., to service as many users as we can) using the w available colors (i,e., using the w available frequencies). When only one frequency is available ($w = 1$) the call control problem is equivalent to the maximum independent set problem.

So, as far as the frequency allocation problem in wireless (cellular) networks is concerned, what we actually trying to achieve is to optimally solve the corresponding minimum coloring problem in the underlying interference graph of the network. On the other hand, in the case of the call control problem on wireless (cellular) networks, we are trying to achieve optimal solutions to the maximum independent set problem in the underlying interference graph of the network. We study the on-line versions of the maximum independent set and the coloring problems. This implies that the graph is not known in advance but its nones appear gradually, and every time a new node appears, our algorithm must make decide, without any knowledge of which will be the next node presented, whether to assign the node one of the colors already used or to use a new additional color, or whether to include the node in the current independent set, or to construct a new one.

1.3 Performance evaluation of on–line algorithms

Competitive analysis [54] has been used for evaluating the performance of on–line algorithms for various problems. In our setting, given a sequence of calls, the performance of an on–line algorithm A is compared to the performance of the optimal algorithm OPT.

For frequency allocation algorithms, let $C_A(\sigma)$ be the cost of the on–line algorithm A on the sequence of calls σ, i.e. the number of frequencies used by A, and $C_{OPT}(\sigma)$ the cost of the optimal algorithm OPT. If A is a deterministic algorithm, we define its competitive ratio ρ as

$$\rho = \max_{\sigma} \frac{C_A(\sigma)}{C_{OPT}(\sigma)},$$

where the maximum is taken over all possible sequences of calls. If A is a randomized algorithm, we define its competitive ratio ρ as

$$\rho = \max_{\sigma} \frac{\mathcal{E}[C_A(\sigma)]}{C_{OPT}(\sigma)},$$

where $\mathcal{E}[C_A(\sigma)]$ is the expectation of the number of frequencies used by A, and the maximum is taken over all possible sequences of calls.

For call control algorithms, let $B_A(\sigma)$ be the benefit of the on–line algorithm A on the sequence of calls σ, i.e. the number of calls of σ accepted by A, and $B_{OPT}(\sigma)$ the benefit of the optimal algorithm OPT. If A is a deterministic algorithm, we define its competitive ratio ρ as

$$\rho = \max_{\sigma} \frac{B_{OPT}(\sigma)}{B_A(\sigma)},$$

where the maximum is taken over all possible sequences of calls. If A is a randomized algorithm, we define its competitive ratio ρ as

$$\rho = \max_{\sigma} \frac{B_{OPT}(\sigma)}{\mathcal{E}[B_A(\sigma)]},$$

where $\mathcal{E}[B_A(\sigma)]$ is the expectation of the number of calls accepted by A, and the maximum is taken over all possible sequences of calls.

Usually, we compare the performance of deterministic algorithms against *off–line adversaries*, i.e. adversaries that have knowledge of the behaviour of the deterministic algorithm in advance. In the case of randomized algorithms, we consider *oblivious adversaries* whose knowledge is limited to the probability distribution of the random choices of the randomized algorithm.

1.4 Contribution of this PhD Thesis

In this PhD Thesis, we study the on-line version of the frequency allocation and the call control problem using competitive analysis. We present new algorithms and lower bounds for wireless networks with cellular, planar and arbitrary interference graphs.

For the frequency allocation problem in cellular networks of reuse distance 2, we improve the best known competitive ratio which had been proved to be at least 3 and achieved by the Fixed Allocation algorithm. In particular, using competitive analysis for the greedy algorithm, we proved that its competitive ratio is at least 2.429 and at most 2.5. These results are presented in Chapter 2.

For the on-line version of the call control problem, we present algorithm p-random, a randomized algorithm that uses randomness whenever a communication request appears (i.e., proportional to the size of the network) in wireless networks of reuse distance 2 that support one frequency. We prove that it achieves a competitive ratio between $2,469$ and $2,651$ assuming oblivious adversaries. In this way, we beat the lower bound of 3, which is the smallest possible competitive ratio obtained by deterministic algorithms in wireless cellular networks. Our analysis can be extended to cellular networks of constant or arbitrary degree. Furthermore, we significantly improve this result presenting a series of simple randomized algorithms which obtain competitive ratios less than 3, work for networks that support arbitrarily many frequencies and, use only either a constant number of random bits or a weak random source. The best upper bound on competitiveness we proved is 7/3.

For cellular networks of reuse distance $k > 2$, we presented simple randomized on-line call control algorithms with competitive ratios that significantly improve the lower bounds of deterministic algorithms and use only $O(\log k)$ random bits or weak random sources. These algorithms achieve competitive ratios significantly better than 4. Furthermore, using Yao's Minimax Principle, we proved new lower bounds of 25/12, 127/60, and $2,5$ on the competitiveness of on-line call control algorithms in cellular networks of reuse distance $k = 2, 3, 4$, $k = 5$ and $k \geq 6$, respectively and a new lower bound of 2.086 for networks with planar interference graphs.

Our results for the call control problem are presented in Chapter 3. In Chapter 4, we study the on-line versions of the two fundamental graph-theoretic problems, the maximum independent set problem and the graph coloring problem, for disk graphs, which are graphs resulting from the intersection of disks on the plane. For the maximum independent set problem, we examine if the use of randomness can help in the improvement of the competitive ratios of on-line algorithms. We assume that the sequences of disks are constructed by oblivious adversaries, i.e., adversaries that do not have knowledge of the random choices of the algorithm (but, they probably know the probability distribution according to which the algorithm makes its random choices). We prove that, in general, the use of randomness does not help against oblivious adversaries even when the disk representation is given as part of the input, i.e., we constructed sequences of disks for which no (possibly randomized) on-line algorithm can have a competitive ratio better than $\Omega(n)$. When the disk representation is not given as part of the input, we prove a lower bound of $\Omega(\min\{n, \sigma^2\})$ on the competitiveness of on-line algorithms in σ-bounded disk graphs (i.e., graphs whose nodes correspond to disks with radii between 1 and σ) with n vertices which implies that algorithm First-Fit is optimal within a constant factor. For σ-bounded disk graphs when the disk representation is given as part of the input, we present randomized algorithms with competitive ratios almost logarithmic in σ and prove that these algorithms are optimal. For unit disk graphs (i.e., graphs of disks of the same radius), we present a randomized algorithm with competitive ratio equal to 4.41 (which is less than the lower bound of 5 on the competitiveness of deterministic algorithms). We also prove lower bounds of 2.5 and 3 on the competitiveness of randomized algorithms for unit disk graphs. For the coloring problem, we show how the best known upper bound of $O(\min\{\log n, \log \sigma\})$ for σ-bounded sequences of n disks can be achieved even if the disk representation is not given as part of the input.

We conclude in Chapter 5 indicating possible directions for future research on the problems we study.

Chapter 2

Frequency Allocation in Cellular Networks

In this chapter, we study the frequency allocation problem in cellular networks. We address the on-line version of the problem.

Solutions to the on-line version of the frequency allocation problem imply the design of on-line protocols for frequency allocation in cellular networks so that signal interference is avoided under the assumption that the number of calls in the cells of the network changes with time.

The static version of the frequency allocation problem including the signal interference constraints can be described as follows. Let $G = (V, E, w)$ be an interference graph where each vertex $v \in V$ is associated with a non-negative integer weight, $w(v) \geq 0$. Graph G models a static instance of the network, its vertices represent cells while the weights represent the number of calls in each cell to be served. The problem is essentially the proper multicoloring of graph G, i.e., the assignment of $w(v)$ distinct frequencies to each vertex v so that for each edge, $(u, v) \in E$, the sets of colors assigned to its endpoints u and v, are disjoint. We define as cost or span of the multicoloring the total number of the colors used. In particular, the objective is the proper multicoloring of G with span equal to the minimum number of colors required to color graph G, which is denoted by $\chi(G)$. In the context of the frequency allocation problem, a multicoloring as defined before, reflects the necessary signal interference constraints: each color stands for a different frequency and the same frequency can be used for two calls iff these calls originate at different, mutually non-adjacent cells. Usually, there is a correspondance beween the available set of colors and the set of non-negative integers. For evaluating the computational complexity of the static version of the frequency allocation problem, we initially define as weight of a maximal clique of G the sum of weights of all vertices of the clique. Obviously, $\chi(G)$ will be at least equal to the weight of the maximal clique of the graph, and since G is a subgraph of the infinite triangular lattice, all possible maximal cliques will be either isolated vertices or edges or triangles.

The problem of optimally multicoloring hexagon graphs has been proved to be $NP-$hard [48]. Regarding upper bounds, there is a wide range of frequency allocation algorithms in the literature according to which few colors are actually needed but there are still no proven bounds on their performance ([18], [40], [41], [53], [60]). The Fixed Allocation algorithm is a very simple algorithm, based on the fact that the underlying graph can be colored with 3 colors. The algorithm uses three predefined, mutually disjoint, color sets, one for each base color and, a vertex of base color 1 uses colors from the first set, while a vertex of base color 2 or 3 uses colors from the second or third set, respectively. The Fixed Allocation algorithm is a 3-approximation algorithm. Janssen et al. in [37] present the Fixed Preference Allocation algorithm which uses at most $\frac{3}{2}$ times the minimum number of colors required. $\frac{4}{3}$-approximation algorithms have been presented in [48] and [49]. In the on-line version of the frequency allocation problem, vertex weights vary (increase) over time. These changes are modelled as a sequence of interference graphs, $G_t = (V, E, w_t) : t \geq 0$, where w_t is the set of calls that must be served at time t. At each time t, an on-line algorithm must decide how to color graph G_t before proceeding with graph G_{t+1} at time $t + 1$. The coloring must be carried out without any knowledge of the graphs already appeared in the sequence.

There has been significant previous work on the on-line version of the graph coloring problem (e.g., [57], [43], [35]), where vertices of the graph to be colored appear gradually, one by one, in each time step, an the algorithm must assign them a color.

The static version of the problem has been studied in [49] and [48], while in [38] the on–line version of the problem is addressed. Janssen et al. in [38] prove lower bounds using competitive analysis for the performance evaluation of several frequency allocation algorithms. Among other results, they prove that no on–line deterministic frequency allocation algorithm can have a competitive ratio better than 2 while they mention the classic Fixed Allocation algorithm which achieves a competitive ratio of 3.

In this chapter, we improve this competitive ratio presenting a tight competitive analysis for the greedy frequency allocation algorithm (Section 2.2). In particular, we prove that the competitive ratio of the greedy algorithm is between 2.429 and 2.5. A recent better lower bound presented in [50] shows that our analysis is tight.

2.1 The greedy frequency allocation algorithm

The Fixed Allocation algorithm [38] uses the fact that three colors, called base colors, suffice for multicoloring the interference graph of a cellular network. The algorithm uses three fixed sets of colors, one for each base color. A cell of base color 1 uses for its calls frequencies from the first color set while a cell of base color 2 or 3 uses colors from the second or third color set, respectively. It can be easily seen that the number of frequencies used by the Fixed Allocation algorithm is at most three times the number of frequencies that the optimal off–line algorithm would need.

An intuitive deterministic algorithm for the on–line frequency allocation problem is the greedy algorithm described as follows. The algorithm considers the frequencies as positive integers $1, 2, \ldots$. When a new call appears, it is assigned the minimum available frequency so that the call does not interfere (is not assigned the same frequency) with calls of the same or adjacent cells. In this section we present an almost tight competitive analysis of the greedy algorithm.

2.2 Analysis of the greedy algorithm

In this section, we present an upper bound for the competitiveness of the greedy algorithm for the frequency allocation problem. In particular:

Theorem 1. *The greedy frequency allocation algorithm is at most* 2.5*–competitive against an off–line adversary when applied to a cellular network.*

Proof. Let σ be a sequence of calls in a cellular network and D be the maximum number of calls in any three mutually adjacent cells of the network. Obviously, D is a lower bound on the number of frequencies required for an optimal frequency allocation to σ.

Consider the execution of the greedy algorithm on σ, and let c_0 be the cell that contains the call that was assigned the highest frequency a_0. We will prove that $a_0 \leq 2.5D$, proving that the greedy algorithm is at most 2.5–competitive against an off–line adversary.

We denote by x_0 the number of calls in cell c_0. By the definition of the greedy algorithm, since the frequency a_0 has been assigned to a call of cell c_0, the frequencies $1, \ldots, a_0 - 1$ must also have been assigned to calls of c_0

and its surrounding cells. Note that the number of all these calls is at most $3D - 2x_0$. Thus, we obtain the following constraint for x_0.

$$x_0 \leq \frac{3D - a_0}{2} \tag{2.1}$$

We call the six surrounding cells of c_0, $c_1, c_2, c_3, c_4, c_5, c_6$, so that for the corresponding highest frequency a_1, a_2, a_3, a_4, a_5, a_6 assigned to calls of these cells, the following inequality holds:

$$a_0 > a_1 \geq a_2 \geq a_3 \geq a_4 \geq a_5 \geq a_6$$

We also denote by $y_i(j)$, for $0 \leq i \neq j \leq 6$, the number of calls in cell c_j that have been assigned higher frequencies than a_i. Obviously, it is $y_i(j) \leq x_j$, for $0 \leq i \neq j \leq 6$.

Consider frequency a_1 assigned to some call of cell c_1. It is $a_1 = a_0 - y_1(0)$. Furthermore, a_1 is upper–bounded by the number of calls in c_1 and its surrounding cells, ignoring the $y_1(0)$ calls of c_0 (which have been assigned frequencies higher than a_1). We obtain that

$$x_1 \leq \frac{3D - a_0}{2} \tag{2.2}$$

For the frequency a_2 assigned to some call of c_2, it is

$$a_2 \geq a_0 - y_2(0) - y_2(1) \tag{2.3}$$

We now distinguish between two main cases.

CASE I: The cell c_2 is adjacent to c_1.

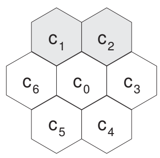

Figure 2.1: Case

In this case , a_2 is upper–bounded by the number of calls in c_2 and its surrounding cells, ignoring the $y_2(0)$ calls of c_0 and the $y_2(1)$ calls of c_1 (which have been assigned frequencies higher than a_2). The number of calls in c_2 and its surrounding cells is at most $2D - x_2 + x_0 + x_1$. Using (3), we obtain that

$$x_2 \leq 2D + x_0 + x_1 - a_0 \tag{2.4}$$

Note that the number of calls in c_0 and its surrounding cells, which is an upper bound for a_0, is at most $2D - x_0 + x_1 + x_2$. Using (2) and (4), we obtain that

$$\begin{aligned} a_0 &\leq 2D - x_0 + x_1 + x_2 \\ &\leq 4D + 2x_1 - a_0 \\ &\leq 7D - 2a_0 \Rightarrow \\ a_0 &\leq \frac{7D}{3}. \end{aligned}$$

CASE II: The cell c_2 is not adjacent to c_1.

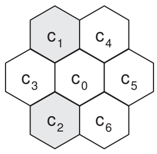

Figure 2.2: Case II

In this case, a_2 is upper–bounded by the number of calls in c_2 and its surrounding cells, ignoring the $y_2(0)$ calls of c_0 (which have been assigned frequencies higher than a_2). The number of calls in c_2 and its surrounding cells is at most $3D - 2x_2$. Using (3), we obtain that

$$x_2 \leq \frac{3D - a_0 + y_2(1)}{2} \tag{2.5}$$

For the frequency a_3 assigned to some call of c_3, it is

$$a_3 \geq a_0 - y_3(0) - y_3(1) - y_3(2) \tag{2.6}$$

We now distinguish between the following three cases.

CASE II.1: The cell c_3 is adjacent to c_1 (and possibly to c_2).

In this case, a_3 is upper–bounded by the number of calls in c_3 and its surrounding cells, ignoring the $y_3(0)$ calls of c_0 and the $y_3(1)$ calls of c_1 (which have been assigned frequencies higher than a_3). The number of calls in c_3 and its surrounding cells is at most $2D - x_3 + x_0 + x_1$. Using (6), we obtain that

$$x_3 \leq 2D + x_0 + x_1 + y_3(2) - a_0 \tag{2.7}$$

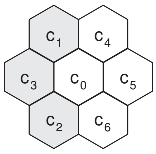

Figure 2.3: Case II.1

Note that the number of calls in c_0 and its surrounding cells, which is an upper bound for a_0, is at most $2D - x_0 + x_1 + x_3$. Using (2), (5), (7), and the fact that $y_3(2) \leq x_2$ and $y_2(1) \leq x_1$, we obtain that

$$\begin{aligned}
a_0 &\leq 2D - x_0 + x_1 + x_3 \\
&\leq 4D + 2x_1 + y_3(2) - a_0 \\
&\leq 4D + 2x_1 + x_2 - a_0 \\
&\leq \frac{11D}{2} + 2x_1 + \frac{y_2(1)}{2} - \frac{3a_0}{2} \\
&\leq \frac{11D}{2} + \frac{5x_1}{2} - \frac{3a_0}{2} \\
&\leq \frac{37D}{4} - \frac{11a_0}{4} \Rightarrow \\
a_0 &\leq \frac{37D}{15}.
\end{aligned}$$

CASE II.2: The cell c_3 is adjacent to c_2 (but not to c_1).

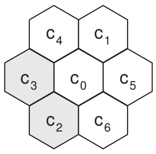

Figure 2.4: Case II.2

In this case, a_3 is upper–bounded by the number of calls in c_3 and its surrounding cells, ignoring the $y_3(0)$ calls of c_0 and the $y_3(2)$ calls of c_2 (which have been assigned frequencies higher than a_3). The number of calls in c_3 and its surrounding cells is at most $2D - x_3 + x_0 + x_2$. Using (6), we obtain that

$$x_3 \leq 2D + x_0 + x_2 + y_3(1) - a_0 \tag{2.8}$$

Note that the number of calls in c_0 and its surrounding cells, which is an upper bound for a_0, is at most $2D - x_0 + x_2 + x_3$. Using (2), (5), (8), and the fact that $y_3(1) \leq x_1$ and $y_2(1) \leq x_1$, we obtain that

$$\begin{aligned}
a_0 &\leq 2D - x_0 + x_2 + x_3 \\
&\leq 4D + 2x_2 + y_3(1) - a_0 \\
&\leq 4D + x_1 + 2x_2 - a_0 \\
&\leq 7D + x_1 + y_2(1) - 2a_0 \\
&\leq 7D + 2x_1 - 2a_0 \\
&\leq 10D - 3a_0 \Rightarrow \\
a_0 &\leq \frac{5D}{2}.
\end{aligned}$$

CASE II.3: The cell c_3 is not adjacent to c_1 and c_2.

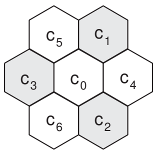

Figure 2.5: Case II.3

In this case, a_3 is upper–bounded by the number of calls in c_3 and its surrounding cells, ignoring the $y_3(0)$ calls of c_0 (which have been assigned frequencies higher than a_3). The number of calls in c_3 and its surrounding cells is at most $3D - 2x_3$. Using (6), we obtain that

$$x_3 \leq \frac{3D - a_0 + y_3(1) + y_3(2)}{2} \tag{2.9}$$

For the frequency a_4 assigned to some call of c_4, it is

$$a_4 \geq a_0 - y_4(0) - y_4(1) - y_4(2) - y_4(3) \tag{2.10}$$

Again, we distinguish between the following three subcases.

CASE II.3.a: The cell c_4 is adjacent to c_1 and c_2.

In this case, a_4 is upper–bounded by the number of calls in c_4 and its surrounding cells, ignoring the $y_4(0)$ calls of c_0, the $y_4(1)$ calls of c_1, and the $y_4(2)$ calls of c_2 (which have been assigned frequencies higher than a_4). The

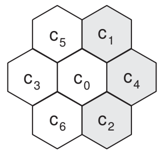

Figure 2.6: Case II.3.a

number of calls in c_4 and its surrounding cells is at most $2D - x_4 + x_0 + x_1$. Using (10), we obtain that

$$x_4 \leq 2D + x_0 + x_1 + y_4(3) - a_0 \tag{2.11}$$

Note that the number of calls in cells c_5 and c_6 which are adjacent to c_3 is $x_5, x_6 \leq D - x_0 - x_3$. Thus, the number of calls in c_0 and its surrounding cells, which is an upper bound for a_0, is at most $2D - x_0 - x_3 + x_1 + x_2 + x_4$. Using (2), (5), (9), (11), and the fact that $y_4(3) \leq x_3$ and $y_2(1) \leq x_1$, we obtain that

$$\begin{aligned}
a_0 &\leq 2D - x_0 - x_3 + x_1 + x_2 + x_4 \\
&\leq 4D - x_3 + 2x_1 + x_2 + y_4(3) - a_0 \\
&\leq \frac{11D}{2} + 2x_1 + \frac{y_2(1)}{2} - \frac{3a_0}{2} \\
&\leq \frac{11D}{2} + \frac{5x_1}{2} - \frac{3a_0}{2} \\
&\leq \frac{37D}{4} - \frac{11a_0}{4} \Rightarrow \\
a_0 &\leq \frac{37D}{15}.
\end{aligned}$$

CASE II.3.b: The cell c_4 is adjacent to c_1 and c_3.

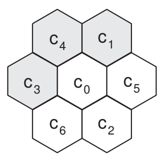

Figure 2.7: Case II.3.b

In this case, a_4 is upper–bounded by the number of calls in c_4 and its surrounding cells, ignoring the $y_4(0)$ calls of c_0, the $y_4(1)$ calls of c_1, and the

$y_4(3)$ calls of c_3 (which have been assigned frequencies higher than a_4). The number of calls in c_4 and its surrounding cells is at most $2D - x_4 + x_0 + x_1$. Using (10), we obtain that

$$x_4 \leq 2D + x_0 + x_1 + y_4(2) - a_0 \tag{2.12}$$

Note that the number of calls in c_0 and its surrounding cells, which is an upper bound for a_0, is at most $2D - x_0 + x_1 + x_4$. Using (2), (5), (9), (12), and the fact that $y_4(2) \leq x_2$ and $y_2(1) \leq x_1$, we obtain that

$$\begin{aligned}
a_0 &\leq 2D - x_0 + x_1 + x_4 \\
&\leq 4D + 2x_1 + y_4(2) - a_0 \\
&\leq 4D + 2x_1 + x_2 - a_0 \\
&\leq \frac{11D}{2} + 2x_1 + \frac{y_2(1)}{2} - \frac{3a_0}{2} \\
&\leq \frac{11D}{2} + \frac{5x_1}{2} - \frac{3a_0}{2} \\
&\leq \frac{37D}{4} - \frac{11a_0}{4} \Rightarrow \\
a_0 &\leq \frac{37D}{15}.
\end{aligned}$$

CASE II.3.c: The cell c_4 is adjacent to c_2 and c_3.

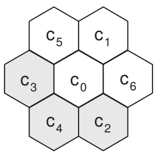

Figure 2.8: Case II.3.c

In this case, a_4 is upper–bounded by the number of calls in c_4 and its surrounding cells, ignoring the $y_4(0)$ calls of c_0, the $y_4(2)$ calls of c_2, and the $y_4(3)$ calls of c_3 (which have been assigned frequencies higher than a_4). The number of calls in c_4 and its surrounding cells is at most $2D - x_4 + x_0 + x_2$. Using (10), we obtain that

$$x_4 \leq 2D + x_0 + x_2 + y_4(1) - a_0 \tag{2.13}$$

Note that the number of calls in c_0 and its surrounding cells, which is an upper bound for a_0, is at most $2D - x_0 + x_2 + x_4$. Using (2), (5), (9), (13), and the fact that $y_4(1) \leq x_1$ and $y_2(1) \leq x_1$, we obtain that

$$\begin{aligned} a_0 &\leq 2D - x_0 + x_2 + x_4 \\ &\leq 4D + 2x_2 + y_4(1) - a_0 \\ &\leq 4D + x_1 + 2x_2 - a_0 \\ &\leq 7D + x_1 + y_2(1) - 2a_0 \\ &\leq 7D + 2x_1 - 2a_0 \\ &\leq 10D - 3a_0 \Rightarrow \\ a_0 &\leq \frac{5D}{2}. \end{aligned}$$

In any case, it is $a_0 \leq 2.5D$. The theorem follows. □

2.3 A lower bound for the competitiveness of the greedy algorithm

We now give a lower bound on the performance of the greedy algorithm. In particular:

Theorem 2. *The greedy algorithm is at least* 2.429*–competitive against an off–line adversary when applied to a cellular network.*

Proof. Consider the cellular network and the sequence of calls σ' shown in the left part of Figure 2.9. Calls that appear in step 1 are assigned frequency 1. At any subsequent step $2 \leq i \leq 17$, the greedy algorithm will assign to all the calls that appear in step i the frequency i, since the frequencies $1, 2, ..., i-1$ are already used by calls in the same or adjacent cells.

In this counterexample, the greedy algorithm uses 17 frequencies while an optimal frequency allocation to σ' with 7 frequencies is depicted in the right part of Figure 2.9. Thus, the competitive ratio of the greedy algorithm is

$$\rho = \max_{\sigma} \frac{C_A(\sigma)}{C_{OPT}(\sigma)} \geq \frac{C_A(\sigma')}{C_{OPT}(\sigma')} = \frac{17}{7} = 2.429.$$

□

In a recent paper [50] Narayanan and Tang presented a better lower bound of 2.5 for the competitiveness of the greedy algorithm, ahich implies that our analysis given in section 2.2 is tight.

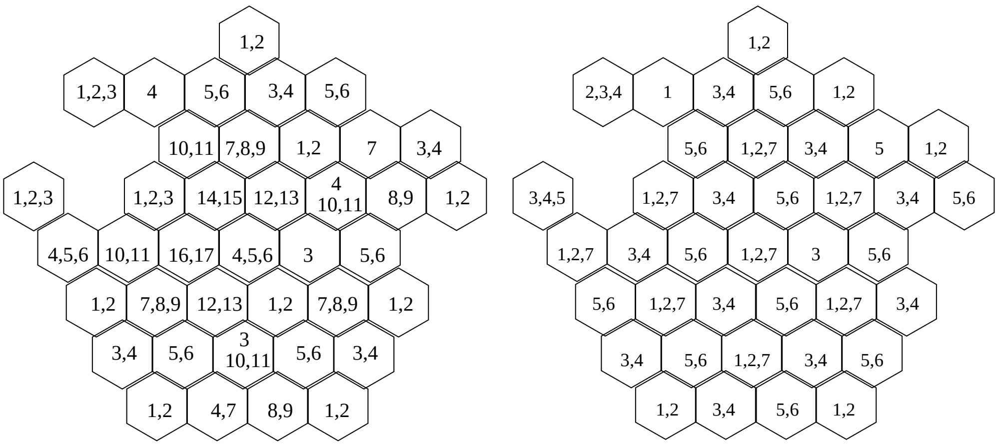

Figure 2.9: The lower bound on the performance of the greedy algorithm. In the left part integers correspond to the step in which a call appears. An optimal allocation of frequencies is depicted in the right part.

Chapter 3

Call Control in Wireless Networks

In this chapter, we study the *call control* (or call admission) problem which is defined as follows:

> Given users that wish to communicate, the *call control* problem on a network that supports a spectrum of w available frequencies is to assign frequencies to users so that signal interference is avoided and the number of users served is maximized.

We assume that calls corresponding to users that wish to communicate appear in the cells of the network in an on–line manner. When a call arrives, a call–control algorithm decides either to accept the call (assigning a frequency to it), or to reject it. Once a call is accepted, it cannot be rejected (preempted). Furthermore, the frequency assigned to the call cannot be changed in the future. We assume that all calls have infinite duration; this assumption is equivalent to considering calls of the same duration.

The static version of the call control problem is very similar to the famous maximum independent set problem. The on–line version of the problem is studied in [1, 3, 8, 12, 42, 52]. [1], [3], and [42] study the call control problem in the context of optical networks. Pantziou et al. [52] present upper bounds for planar and arbitrary mobile networks. Applying the CLASSIFY AND RANDOMLY SELECT paradigm [3, 52] on cellular networks, we obtain a 3–competitive randomized call control algorithm. Usually, competitive analysis of call control focuses on networks supporting one frequency. Awerbuch et al. [1] present a simple way to transform algorithms designed for one frequency to algorithms for arbitrarily many frequencies with a small sacrifice in competitiveness. Lower bounds for call control in arbitrary networks are presented in [8].

The greedy algorithm is probably the simplest on-line algorithm. When a call arrives, the greedy algorithm seeks for the first available frequency. If such a frequency exists, the algorithm accepts the call assigning this frequency to it, otherwise, the call is rejected. In general, Pantziou et al. [52] show that this algorithm has competitive ratio equal to the degree of the interference graph and no better in general. The greedy algorithm is optimal within the class of deterministic on-line call control algorithms.

Simple randomized algorithms can be defined using the "classify and randomly select" paradigm [2, 3, 52]. Such algorithms use a coloring of the underlying interference graph, randomly select a color out of the colors used, and execute the greedy algorithm in the cells colored with the selected color, ignoring (i.e., rejecting) calls in all other cells. The competitive ratio achieved in this way, against oblivious adversaries, is equal to the number of colors used in the coloring of the interference graph.

A detailed description of the greedy call control algorithm as well as of the ""classify and randomly select"" paradigm can be found on Section 3.1.

In cellular networks of reuse distance 2, the greedy algorithm is 3-competitive against off-line adversaries, in the case of one frequency. Slightly worse competitiveness bounds can be proved in the case of arbitrarily many frequencies using the techniques of [2, 21, 59]. In [15], using similar arguments with those of [52], it was observed that no deterministic on-line call control algorithm in cellular networks of reuse distance 2 can be better than 3-competitive against off-line adversaries. Applying the "classify and randomly select" paradigm using a 3-coloring of the interference graph, we obtain a 3-competitive randomized algorithm even in the case of arbitrarily many frequencies. Observe that this algorithm uses a very weak random source which equiprobably selects one out of three distinct objects.

describe algorithm p–RANDOM, an intuitive on–line randomized call control algorithm for networks that support one frequency. They present upper and lower bounds on the competitive ratio of the algorithm as functions of parameter p and, by optimizing these functions, they prove that, for some value of p, the competitive ratio of algorithm p-RANDOM against oblivious adversaries is between 2.469 an 2.651. The analysis of algorithm p-RANDOM in [15] applies only to cellular networks with one frequency but it indicates that randomization helps to beat the deterministic upper bounds. However, the number of random bits used by the algorithm may be proportional to the size of the network. The best known lower bound on the competitive ratio of any randomized call control algorithm in cellular networks of reuse distance 2 is 1.857 [15].

For the on-line version of the call control problem, we describe algorithm p–RANDOM (Section 3.2), an intuitive on–line randomized call control algorithm for networks that support one frequency [12]. We present upper and lower bounds on the competitive ratio of the algorithm as functions of parameter p against oblivious adversaries and, by optimizing these functions, we prove that, for some value of p, the competitive ratio of algorithm p-RANDOM against oblivious adversaries is at most 2.651. In this way, we beat the deterministic lower bound of 3, which is the minimum competitive ratio that can be achieved by deterministic algorithms in wireless cellular networks.

These results [12, 13] hold only for networks that support one frequency but explicitly show that randomness helps in the improvement of upper bounds for deterministic algorithms.

In Section 3.3, we give new analytical bounds on the competitiveness of algorithm p–RANDOM in sparse wireless cellular networks, i.e., wireless cellular networks in which cells may have irregular shape but the network degree is small. We consider networks of degree three and four; for networks of degree three, we prove that for a specific value of p, algorithm p–RANDOM is 9/4–competitive for networks supporting one frequency and 2.787–competitive for networks supporting many frequencies. We also outline the proof for networks of degree 4; for these networks we can achieve competitive ratios of 2.651 and 3.182 for networks supporting one and many frequencies, respectively. Surprisingly, this matches the best known result for ideal wireless cellular networks presented in [13].

Next, we study the on-line version of the call control in cellular networks of reuse distance $k > 2$. The greedy algorithm has competitive ratio 4 and 5 in cellular networks of reuse distance $k \in \{3, 4, 5\}$ and $k \geq 6$, respectively, which support one frequency. This is due to the fact that the acceptance of a non-optimal call may cause the rejection of at most 4 and 5 optimal calls, respectively. These competitive ratios are the best possible that can be achieved by deterministic algorithms. Using the techniques of [2, 21, 59], it can be shown that, in the case of arbitrarily many frequencies, the greedy algorithm has competitive ratio at most 4.521 and at most 5.517 in cellular networks of reuse distance $k \in \{3, 4, 5\}$ and $k \geq 6$, respectively. Furthermore, applying the "classify and randomly select" paradigm using an efficient coloring of the interference graph of cellular networks of distance reuse $k > 2$ would give randomized on-line algorithms with competitive ratio $\Omega(k^2)$. Even in the case of $k = 3$, the competitive ratio we obtain in this way is 7.

In this PhD Thesis, we improve previous results on the competitiveness of on-line call control algorithms in cellular networks (Section 3.6). We present algorithms based on the "classify and randomly select" paradigm which use new colorings of the interference graph. These algorithms use a small number of random bits, and have small competitive ratios against oblivious adversaries even in the case of arbitrarily many frequencies. In particular, in cellular networks of reuse distance 2, we significantly improve the best known competitiveness bounds achieved by algorithm p-RANDOM by presenting a series of simple randomized algorithms that have smaller competitive ratios, work on networks with arbitrarily many frequencies, and use only a constant number of random bits or a comparable weak random source. The best competitiveness upper bound we obtain is 7/3. In cellular networks of reuse distance $k > 2$, we present (Section 3.7) simple randomized on-line call control algorithms with competitive ratios which significantly beat the lower bounds on the competitiveness of deterministic algorithms and use only $O(\log k)$ random bits. For any $k > 2$, the competitive ratio we achieve is strictly smaller than 4.

No deterministic on-line call control algorithm can achieve a competitive ratio better than 3 against off-line adversaries in cellular networks of reuse distance 2, supporting one frequency. In Section 3.8, we extend this lower bound to cellular networks of reuse distance $k \in \{3, 4, 5\}$ and $k \geq 6$ that support one frequency and prove lower bounds of 4 and 5, respectively. For randomized algorithms in cellular networks of reuse distance $k \geq 5$ and $k = 12$, we prove corresponding lower bounds on their competitiveness of 25/12 and 127/60, respectively, against oblivious adversaries. We consider networks that support one frequency; these bounds can be trivially extended to networks that support arbitrarily many frequencies.

3.1 Description of known algorithms

In this section, we briefly describe two well–known on–line algorithms for call control in wireless cellular networks: the greedy algorithm and the CLASSIFY AND RANDOMLY SELECT paradigm. Furthermore, we present a simple way for transforming call control algorithms designed for networks with one frequency to call control algorithms for networks with arbitrarily many frequencies, with a small sacrifice in competitiveness. Also, we present a lower bound on the competitiveness of deterministic on-line call control algorithms.

Assume that a sequence of calls σ appears in a network that support

w frequencies $1, 2, ..., w$. The greedy algorithm is an intuitive deterministic algorithm. For any new call c at a cell v, the greedy algorithm searches for the minimum available frequency, i.e., for the minimum frequency among frequencies $1, 2, ..., w$ that has not been assigned to calls in cell v or its adjacent cells. If such a frequency exists, the call c is accepted and is assigned this frequency; otherwise, the call is rejected.

Pantziou et al. [52] have proved that this algorithm is at most $(\Delta+1)$–competitive against off–line adversaries for networks supporting many frequency (and Δ–competitive for networks supporting one frequency), where Δ is the degree of the network. We extend their technique and prove the following theorem.

Theorem 3. *Let $G = (V, E)$ be an interference graph, v a vertex of G, and Γ_v the maximum independent set in the neighborhood of v. The greedy algorithm is $\frac{1}{1-e^{-\frac{1}{\gamma}}}$–competitive against an off–line adversary, where*

$$\gamma = \max_{v \in V} |\Gamma_v|.$$

Proof. Let B_A be the set of calls accepted by the greedy algorithm and B_{OPT} the set of calls accepted by the optimal algorithm.

The number of calls rejected by A because of the calls in $B_A \backslash B_{OPT}$ is at most $\gamma|B_A \backslash B_{OPT}$. Indeed, the best the optimal algorithm could do is to reject a call $r_i \in B_A \backslash B_{OPT}$ in a cell c_i accepted by the greedy algorithm, and accept at most γ calls that appear in cells adjacent to c_i.

Since the network supports only one frequency, no call in $B_{OPT} \cap B_A$ can cause the rejection of any other call in B_{OPT}, because the calls in B_{OPT} appear in distinct not adjacent cells. Thus,

$$|B_{OPT} \backslash B_A| \leq \gamma |B_A \backslash B_{OPT}| \Rightarrow$$

$$|B_{OPT} \backslash B_A| + |B_{OPT} \cap B_A| \leq \gamma |B_A \backslash B_{OPT}| + |B_{OPT} \cap B_A| \Rightarrow$$

$$|B_{OPT}| \leq \gamma |B_A|.$$

The lemma follows.

□

Awerbuch et al. in [1] present a simple way for transforming call control algorithms designed for networks with one frequency to call control algorithms for networks with arbitrarily many frequencies, with a small sacrifice in competitiveness. Consider a wireless cellular network and a (deterministic or randomized) on–line call control algorithm ALG–1 for one frequency.

A call control algorithm ALG for w frequencies can be constructed in the following way. For each call c, we execute the algorithm ALG–1 for each of the w frequencies until either c is accepted or the frequency spectrum is exhausted (and the call c is rejected), i.e.,

```
for any new call c
    for i = 1 to w
        run ALG–1(c) for frequency i
        if c was accepted then
            assign frequency i to c
            stop
    reject c.
```

Awerbuch et al. in [1] prove that if ALG–1 is ρ–competitive, then ALG has competitive ratio

$$\frac{1}{1 - \exp(-1/\rho)}.$$

In this way, we can prove that the greedy algorithm in networks that support arbitrarily many frequencies achieves competitive ratio equal to

$$\frac{1}{1 - \exp(-1/\gamma)} < \Delta + 1,$$

using the fact that the greedy algorithm for one frequency is γ–competitive.

For cellular networks, where the interference graph is a hexagon graph, it is $\gamma = 3$, and theorem 3 yields the following corollary.

Corollary 4. *The greedy algorithm is* 3.53*–competitive against an off–line adversary when applied to cellular networks.*

The Classify and Randomly Select paradigm uses a coloring of the cells of the network (coloring of the interference graph) with positive integer (colors) $1, 2, ...$ in such way that adjacent cells are assigned different colors. The randomized algorithm classifies the calls of the sequence into a number of classes; class i contains calls appeared in cells colored with color i. It then selects uniformly at random one of the classes, and considers only calls that belong to the selected class, rejecting all other calls. Once a call of the selected class appears, the greedy algorithm is used.

Using simple arguments, Awerbuch et al. in [3] (see also Pantziou et al. [52]) prove that the Classify and Randomly Select algorithm is χ–competitive against oblivious adversaries, where χ is the number of colors used in the coloring of the cells of the network. This may lead to algorithms with competitive ratio equal to the chromatic number (and no better, in general) of the corresponding interference graph, given that an optimal coloring (i.e., with the minimum number of colors) is available. Note that, in wireless cellular networks of degree Δ, the chromatic number of the corresponding interference graph may be up to $\Delta + 1$.

Theorem 5 ([1, 52])**.** *Algorithm A_{CRS} is 3-competitive against oblivious adversaries.*

note that γ is a lower bound for the competitive ratio of every deterministic algorithm. Consider a network that supports one frequency and is composed of a cell v and γ mutually non-adjacent cells $v_1, v_2, \cdots, v_\gamma$ which are adjacent to v. Consider, now, the following sequence of calls produced by an adversary that has knowledge of the way that the algorithm makes its decisions. First, a call c is presented in cell v. If the algorithm rejects c, then the adversary stops the sequence. In this case, the algorithm has no benefit from its execution. If the algorithm accepts the call c, the adversary presents γ calls c_1, c_2, ..., c_γ in cells v_1, v_2, ..., v_γ, respectively. The benefit of the algorithm is then 1 while the optimal algorithm would obtain benefit γ by rejecting call c and accepting the calls c_1, ..., c_γ.

For cellular networks, the following theorem holds:

Theorem 6. *No deterministic algorithm can be better than 3-competitive against an off-line adversary.*

Obvisouly, the best the algorithm A can do is to accept all calls presented in cells which are non-adjacent to cells where a previously accepted calls are located. But this is exactly what the greedy algorithm does for networks that support one frequency.

Recent work of Trevisan [55] on the maximum independent set problem on bounded–degree graphs implies that, in general, the static version of the call control problem on networks of degree Δ is inapproximable within $O\left(\Delta/2^{O\left(\sqrt{\log \Delta}\right)}\right)$. This means that practical (i.e., algorithms which make their decisions in polynomial time) on-line randomized algorithms with competitive ratio asymptotically better than $O(\Delta^{1-\epsilon})$ for some $\epsilon > 0$ are infeasible.

3.2 Algorithm p–Random

In this section, we present and analyze the algorithm p–Random, a randomized call control algorithm for cellular networks that support one frequency. Algorithm p–Random receives as input a sequence of calls in an on–line manner, and works as follows.

1.Initially, all cells are unmarked.
2.for any new call c in a cell v
3.   if v is marked then reject c.
4.   if v has an accepted call or is adjacent to a cell
      with an accepted call, then reject c
5.   else
6.   with probability p accept c.
7.   with probability $1-p$ reject c and mark v.

The algorithm uses a parameter $p \in [1/\Delta, 1]$, where Δ is the degree of the network. Obviously, if it is $p < 1/\Delta$, the competitive ratio will be greater than Δ, since the expected benefit of the algorithm on a sequence of a single call will be p. The algorithm is simple and can be easily implemented with small communication overhead (exchange of messages) between the base stations of the network.

Marking cells on rejection guarantees that algorithm p–random does not simulate the greedy deterministic one. Assume otherwise, that marking is not used. Then, consider an adversary that presents t calls in a cell v and one call in Δ (mutually not adjacent) cells adjacent to v. The probability that the randomized algorithm does not accept a call in cell v drops exponentially as t increases, and the benefit approaches 1, while the optimal benefit is Δ.

Note that algorithm p–Random may accept at most one call in each cell but this is also the case for any algorithm running in networks that support one frequency (including the optimal one). Thus, for the competitive analysis of algorithm p–Random, we will only consider sequences of calls with at most one call per cell. Also, there is no need for taking into account the procedure of marking cells during the analysis.

We now prove the upper bound on the competitive ratio of algorithm p–Random as a function of p. Our main statement is the following.

Theorem 7. *For $p \in [1/3, 1]$, algorithm p–random has competitive ratio at most*

$$\frac{3}{5p - 7p^2 + 3p^3}$$

against oblivious adversaries.

Proof. Let σ be a sequence of calls. We assume that σ has been fixed in advance and will be revealed to the algorithm in an on–line manner. We make this assumption because we are interested in the competitiveness of the algorithm against oblivious adversaries whose knowledge is limited to the probability distribution of the random choices of the algorithm (i.e., the parameter p).

Consider the execution of algorithm p–RANDOM on σ. For any call $c \in \sigma$, we denote by $X(c)$ the random variable that indicates whether the algorithm accepted c. Clearly, the benefit of algorithm p–RANDOM on σ can be expressed as

$$B(\sigma) = \sum_{c \in \sigma} X(c).$$

Let $A(\sigma)$ be the set of calls in σ accepted by the optimal algorithm. For each call $c \in A(\sigma)$, we define the amortized benefit $\bar{b}(c)$ as

$$\bar{b}(c) = X(c) + \sum_{c' \in \gamma(c)} \frac{X(c')}{d(c')},$$

where $\gamma(c)$ denotes the set of calls of the sequence in cells adjacent to c. For each call $c' \notin A(\sigma)$, $d(c')$ is the number of calls in $A(\sigma)$ that are in cells adjacent to the cell of c. By the two equalities above, it is clear that

$$B(\sigma) = \sum_{c \in A(\sigma)} \bar{b}(c).$$

Furthermore, note that for any call $c' \notin A(\sigma)$, $d(c') \leq 3$. We obtain that

$$\bar{b}(c) \geq X(c) + \frac{\sum_{c' \in \gamma(c)} X(c')}{3}$$

and, by linearity of expectation,

$$\mathcal{E}[B(\sigma)] \geq \sum_{c \in A(\sigma)} \left(\mathcal{E} \left[X(c) + \frac{\sum_{c' \in \gamma(c)} X(c')}{3} \right] \right) \tag{3.1}$$

Let $\gamma'(c)$ be the set of calls in cells adjacent to the cell of c which appear prior to c in the sequence σ. Clearly, $\gamma'(c) \subseteq \gamma(c)$, which implies that

$$\sum_{c' \in \gamma(c)} X(c') \geq \sum_{c' \in \gamma'(c)} X(c').$$

Thus, (3.14) yields

$$\mathcal{E}[B(\sigma)] \geq \sum_{c \in A(\sigma)} \left(\mathcal{E} \left[X(c) + \frac{\sum_{c' \in \gamma'(c)} X(c')}{3} \right] \right) \tag{3.2}$$

In what follows we will try to bound from below the expectation of the random variable $X(c) + \frac{\sum_{c' \in \gamma'(c)} X(c')}{3}$, for each call $c \in A(\sigma)$.

We concentrate on a call $c \in A(\sigma)$. Let $\Omega = 2^{\gamma'(c)}$ be the set which contains all possible subsets of $\gamma'(c)$. We define the *effective neighborhood* of c, denoted by $\Gamma(c)$, to be the subset of $\gamma'(c)$ that contains the calls of $\gamma'(c)$ which, when they appear, they are unconstrained by calls of σ at distance 2 from c. Clearly, $\Gamma(c)$ is a random variable taking its values from the sample space Ω. Intuitively, whether an optimal call c is accepted by the algorithm depends on its effective neighborhood $\Gamma(c)$. We have

$$\begin{aligned}
&\mathcal{E}\left[X(c) + \frac{\sum_{c' \in \gamma'(c)} X(c')}{3}\right] = \\
&\sum_{\gamma \in \Omega} \mathcal{E}\left[X(c) + \frac{\sum_{c' \in \gamma} X(c')}{3} | \Gamma(c) = \gamma\right] \cdot \Pr[\Gamma(c) = \gamma] \geq \\
&\min_{\gamma \in \Omega} \left\{ \mathcal{E}\left[X(c) + \frac{\sum_{c' \in \gamma} X(c')}{3} | \Gamma(c) = \gamma\right] \right\} = \\
&\min_{\gamma \in \Omega} \left\{ \mathcal{E}[X(c) | \Gamma(c) = \gamma] + \frac{\mathcal{E}[\sum_{c' \in \gamma} X(c') | \Gamma(c) = \gamma]}{3} \right\} \qquad (3.3)
\end{aligned}$$

To compute $\mathcal{E}[X(c)|\Gamma(c) = \gamma]$, we observe that algorithm p–Random may accept c only if it has rejected all calls in its effective neighborhood γ. The probability that all calls of γ are rejected given that $\Gamma(c) = \gamma$ is $(1-p)^{|\gamma|}$, and then c is accepted with probability p. Thus,

$$\mathcal{E}[X(c)|\Gamma(c) = \gamma] = p(1-p)^{|\gamma|}. \qquad (3.4)$$

We now bound from below $\mathcal{E}[\sum_{c' \in \gamma} X(c')|\Gamma(c) = \gamma]$ by distinguishing between cases according to the size of the effective neighborhood $|\gamma|$.

Claim 8. *For all $p \in [1/3, 1]$,*

$$\mathcal{E}\left[\sum_{c' \in \gamma} X(c') | \Gamma(c) = \gamma\right] \geq \begin{cases} 0 & \textit{if } |\gamma| = 0 \\ p & \textit{if } |\gamma| = 1 \\ 2p - p^2 & \textit{if } |\gamma| = 2 \\ 3p - 2p^2 & \textit{if } |\gamma| = 3 \\ 4p - 3p^2 + p^3 & \textit{if } |\gamma| = 4 \\ 5p - 4p^2 + p^3 & \textit{if } |\gamma| = 5 \\ 6p - 5p^2 + p^3 & \textit{if } |\gamma| = 6 \end{cases}$$

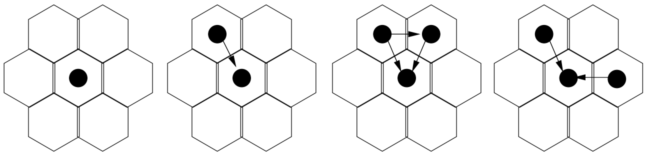

Figure 3.1: The cases $|\gamma| = 0, 1, 2$.

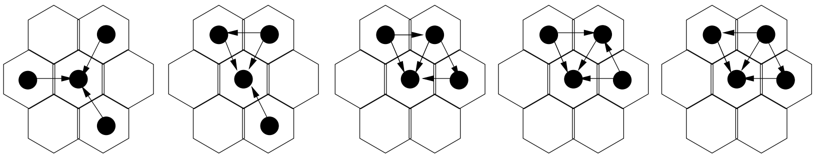

Figure 3.2: The case $|\gamma| = 3$.

Proof. In Figures 3.1, 3.2, 3.3, 3.4, and 3.5 we give all possible cases for the effective neighborhood of an optimal call c in a sequence of calls σ. In each figure the optimal call is denoted by the black circle in the middle cell while black circles in the outer cells denote calls in the effective neighborhood γ of c. An arrow from a call c_1 to another call c_2 indicates that c_1 appears in σ prior to c_2. In the figures, we have eliminated the symmetric cases.

The proof is trivial for the cases $|\gamma| = 0$ and $|\gamma| = 1$ (the two leftmost cases in Figure 3.1). In the third case of Figure 3.1 (where $|\gamma| = 2$), we observe that the algorithm accepts the first call in γ with probability p and the second one with probability $p(1 - p)$. In total, the expectation of the number of accepted calls in γ is $2p - p^2$. In the rightmost case of Figure 3.1, the expectation of the number of accepted calls in γ is $2p < 2p - p^2$.

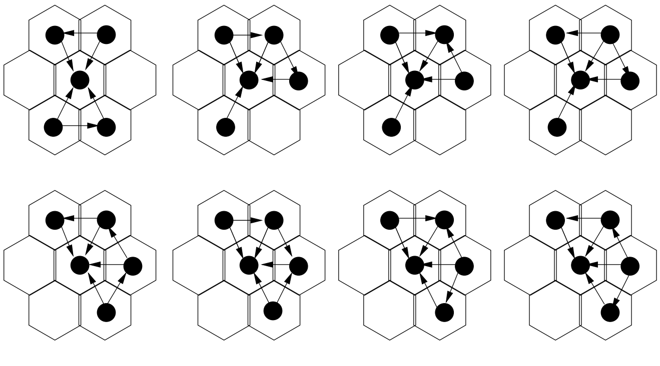

Figure 3.3: The case $|\gamma| = 4$.

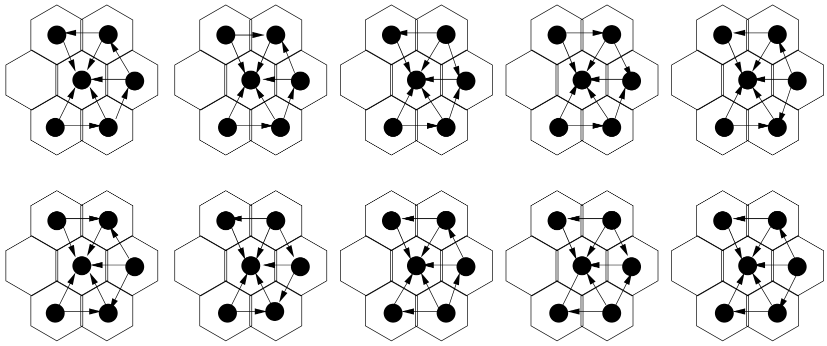

Figure 3.4: The case $|\gamma| = 5$.

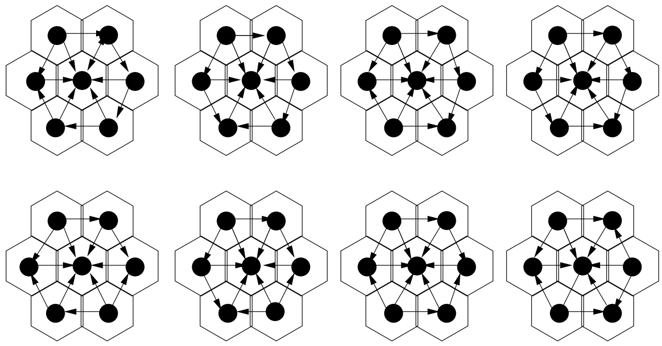

Figure 3.5: The case $|\gamma| = 6$.

Similarly, we can compute the desired lower bounds on $\mathcal{E}[\sum_{c' \in \gamma} X(c')|\Gamma(c) = \gamma]$ for the cases $|\gamma| = 3, 4, 5, 6$. $\square$

By making calculations with (3.16), (3.17), and Claim 8, we obtain that

$$\mathcal{E}\left[X(c) + \frac{\sum_{c' \in \gamma'(c)} X(c')}{3}\right] \geq$$

$$\min_{\gamma \in \Omega}\left\{\mathcal{E}[X(c)|\Gamma(c) = \gamma] + \frac{\mathcal{E}[\sum_{c' \in \gamma} X(c')|\Gamma(c) = \gamma]}{3}\right\} \geq$$

$$\min_{\gamma \in \Omega : |\gamma| = 2}\left\{\mathcal{E}[X(c)|\Gamma(c) = \gamma] + \frac{\mathcal{E}\left[\sum_{c' \in \gamma} X(c')|\Gamma(c) = \gamma\right]}{3}\right\} \geq$$

$$p(1-p)^2 + \frac{2p - p^2}{3} = \frac{5p - 7p^2 + 3p^3}{3}$$

Now, using (3.15) we obtain that

$$\begin{aligned}\mathcal{E}[B(\sigma)] &\geq \sum_{c\in A(\sigma)} \frac{5p-7p^2+3p^3}{3} \\ &= \frac{5p-7p^2+3p^3}{3} \cdot B_{OPT}(\sigma)\end{aligned}$$

This completes the proof of Theorem 7. □

The expression in Theorem 7 is minimized to $729/265 = 2.651$ for $p = 5/9$. Thus, we obtain the following result.

Corollary 9. *There exists an on–line randomized call control algorithm for cellular networks with one frequency which is at most* 2.651*–competitive against oblivious adversaries.*

In the following, we show that our analysis is not far from being tight. In particular, we prove the following.

Theorem 10. *The competitive ratio of algorithm* p–Random *against oblivious adversaries is at least*

$$\max\left\{\frac{3}{4p-3p^2}, \frac{3}{5p-7p^2+4p^3-p^4}\right\}.$$

Proof. We will prove the lower bound by constructing two sequences σ_1 and σ_2 of calls for which the competitive ratio of algorithm p–Random is $\frac{3}{4p-3p^2}$ and $\frac{3}{5p-7p^2+4p^3-p^4}$, respectively.

Sequence σ_1 is depicted in the left part of Figure 3.6. In round 1, a call appears at some cell c, and in round 2 one call appears in each one of the three mutually adjacent cells in the neighborhood of c. Clearly, the benefit of the optimal algorithm is 3. To compute the expectation of the benefit of algorithm p–Random, we observe that with probability p, the call presented in round 1 is accepted, and with probability $1-p$, the call presented in round 1 is rejected and each of the three calls presented in round 2 is accepted with probability p. Thus, the expectation of the benefit of the algorithm on sequence σ_1 is $p + (1-p)3p = 4p - 3p^2$.

Sequence σ_2 is depicted in the right part of Figure 3.6. Calls appear in four rounds. The labels on the calls denote the round in which the calls appear. Clearly, the benefit of the optimal algorithm is 18 since the optimal algorithm would accept the calls which appear in rounds 3 and 4. To compute the expectation of the benefit of algorithm p–Random on sequence σ_2, we first compute the probability that each call is accepted.

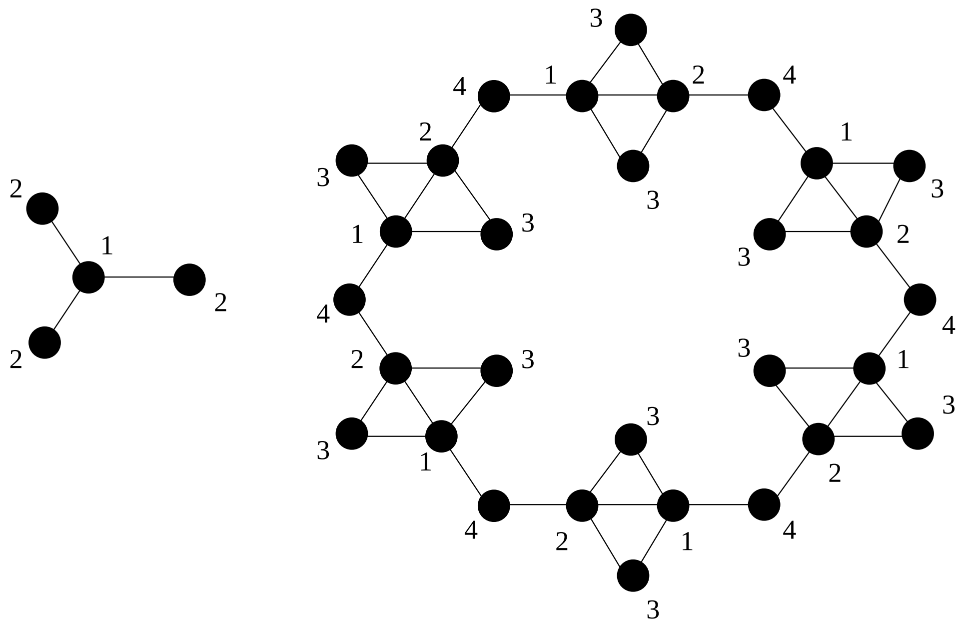

Figure 3.6: The lower bound on the performance of algorithm p–Random.

- A call which appears in round 1 is accepted with probability p.
- A call which appears in round 2 can be accepted if its adjacent call which appeared in round 1 has been rejected; thus, the probability that a call which appears in round 2 is accepted is $p(1-p)$.
- A call which appears in round 3 can be accepted if its adjacent calls which appeared in rounds 1 and 2 have been rejected; thus, the probability that a call which appears in round 3 is accepted is $p(1-p)^2$.
- A call which appears in round 4 can be accepted if its adjacent calls which appeared in rounds 1 and 2 have been rejected. The probability that a call which appears in round 1 is rejected is $1-p$ while the probability that a call which appears in round 2 is rejected is $1-p-(1-p)^2$. Thus, the probability that a call which appears in round 4 is accepted is $p(1-p)\left(1-p-(1-p)^2\right)$.

Note that the number of calls that appear in rounds 1, 2, 3, and 4 is 6, 6, 12, and 6, respectively. Thus, we obtain that the expectation of the benefit of the algorithm is

$$6p+6p(1-p)+12p(1-p)^2+6p(1-p)\left(1-p-(1-p)^2\right) = 5p-7p^2+4p^3-p^4.$$

This completes the proof of the theorem. $\square$

The expression in Theorem 10 is minimized for $p \approx 0.6145$ to 2.469. Thus, we obtain the following corollary.

Corollary 11. *For any* $p \in (1/3, 1)$*, algorithm* p–Random *is at least* 2.469–*competitive against oblivious adversaries.*

Let $p^* \in (1/3, 1)$ be that value such that algorithm p^*–Random has better competitive ratio than any algorithm p–Random, for $p \in [1/3, 1]$. By Theorem 10 and Corollary 9, solving the inequality

$$\max\left\{\frac{3}{4p - 3p^2}, \frac{3}{5p - 7p^2 + 4p^3 - p^4}\right\} \leq \frac{729}{275}$$

we obtain the following.

Corollary 12. $p^* \in [0.421, 0.843]$.

Discussion

We have computed upper and lower bounds on the competitiveness of algorithm p–Random in cellular networks that support one frequency as a function of p. A graphical representation of both functions is depicted in Figure 3.7. Note that there is still a small gap between the upper and the lower bound which is up to 0.25 for some values of p.

So far, we have addressed the case of networks supporting one frequency. A call control algorithm for w frequencies can be constructed in the following way. For each call c, we execute the algorithm p–Random for each of the w frequencies until either c is accepted or the frequency spectrum is exhausted (and the call c is rejected). Using this technique (which is analyzed by Awerbuch et al. in [1]), we obtain a call control algorithm for networks with w frequencies with competitive ratio

$$\frac{1}{1 - \left(1 - \frac{1}{w\rho(p)}\right)^w}$$

where $\rho(p)$ is the competitive ratio of algorithm p–Random in networks supporting one frequency. The competitive ratio we achieve in this way for two frequencies is 2.927. Unfortunately, for larger numbers of frequencies, the competitive ratio we obtain using the same technique is larger than 3.

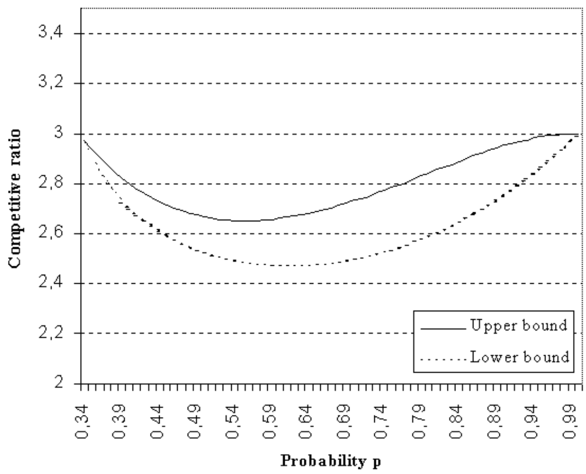

Figure 3.7: A graphical representation of the the upper and the lower bound on the competitiveness of algorithm p–Random for $p \in [1/3, 1]$.

3.3 Analysis of algorithm p-Random in sparse networks

In this section we provide the analysis of the competitiveness of algorithm p–Random. We first present the analysis which is not network specific. Then, using our analysis, we prove the upper bound on the competitive ratio of the algorithm for networks of degree three (in Section 3.3.1) and, we outline the proof of the upper bound for networks of degree four (in Section 3.3.2).

Let σ be a sequence of calls. We assume that σ has been fixed in advance and will be revealed to the algorithm in an on–line manner. We make this assumption because we are interested in the competitiveness of the algorithm against oblivious adversaries whose knowledge is limited to the probability distribution of the random choices of the algorithm (i.e., the parameter p).

Consider the execution of algorithm p–Random (for some $p \in [0, 1]$) on σ. For any call $c \in \sigma$, we denote by $X(c)$ the random variable that indicates whether the algorithm accepted c, i.e.,

$$X(c) = \begin{cases} 0 & \text{if c is rejected} \\ 1 & \text{if c is accepted} \end{cases}$$

Obviously,

$$B(\sigma) = \sum_{c\in\sigma} X(c).$$

Let $A(\sigma)$ be the set of calls in σ accepted by the optimal algorithm. For each call $c \in A(\sigma)$, we define the amortized benefit $\bar{b}(c)$ as

$$\bar{b}(c) = X(c) + \sum_{c'\in\gamma(c)} \frac{X(c')}{d(c')},$$

where $\gamma(c)$ denotes the set of calls of the sequence in cells adjacent to c. For each call $c' \notin A(\sigma)$, $d(c')$ is the number of calls in $A(\sigma)$ that are in cells adjacent to the cell of c'. By the two equalities above, it is clear that

$$B(\sigma) = \sum_{c\in A(\sigma)} \bar{b}(c).$$

Thus, by linearity of expectation,

$$\mathcal{E}[B(\sigma)] = \sum_{c\in A(\sigma)} \left(\mathcal{E}\left[X(c) + \sum_{c'\in\gamma(c)} \frac{X(c')}{d(c')} \right] \right) \tag{3.5}$$

Let $\gamma'(c)$ be the set of calls in cells adjacent to the cell of c which appear prior to c in the sequence σ. Clearly, $\gamma'(c) \subseteq \gamma(c)$, which implies that

$$\sum_{c'\in\gamma(c)} \frac{X(c')}{d(c')} \geq \sum_{c'\in\gamma'(c)} \frac{X(c')}{d(c')}.$$

Thus, (1) yields

$$\begin{aligned} \mathcal{E}[B(\sigma)] &\geq \sum_{c\in A(\sigma)} \left(\mathcal{E}\left[X(c) + \sum_{c'\in\gamma'(c)} \frac{X(c')}{d(c')} \right] \right) \\ &\geq |A(\sigma)| \min_{c\in A(\sigma)} \left\{ \mathcal{E}\left[X(c) + \sum_{c'\in\gamma'(c)} \frac{X(c')}{d(c')} \right] \right\} \end{aligned} \tag{3.6}$$

In what follows we will try to bound from below the expectation of the random variable

$$Y(c) = X(c) + \sum_{c'\in\gamma'(c)} \frac{X(c')}{d(c')},$$

for each call $c \in A(\sigma)$.

We concentrate on a call $c \in A(\sigma)$. Let $\Omega = 2^{\gamma'(c)}$ be the set which contains all possible subsets of $\gamma'(c)$. We define the *effective neighborhood* of c, denoted by $\Gamma(c)$, to be the subset of $\gamma'(c)$ that contains the calls of $\gamma'(c)$ which, when they appear, they are unconstrained by calls of σ at distance 2 from c. Clearly, $\Gamma(c)$ is a random variable taking its values from the sample space Ω. Intuitively, whether an optimal call c is accepted by the algorithm depends on its effective neighborhood $\Gamma(c)$. We have

$$\begin{aligned}
\mathcal{E}[Y(c)] &= \sum_{\gamma\in\Omega}\left(\mathcal{E}\left[X(c)+\sum_{c'\in\gamma}\frac{X(c')}{d(c')}|\Gamma(c)=\gamma\right]\cdot \Pr[\Gamma(c)=\gamma]\right)\\
&\geq \min_{\gamma\in\Omega}\left\{\mathcal{E}\left[X(c)+\sum_{c'\in\gamma}\frac{X(c')}{d(c')}|\Gamma(c)=\gamma\right]\right\}\\
&= \min_{\gamma\in\Omega}\left\{\mathcal{E}[X(c)|\Gamma(c)=\gamma]+\mathcal{E}\left[\sum_{c'\in\gamma}\frac{X(c')}{d(c')}|\Gamma(c)=\gamma\right]\right\} \quad (3.7)
\end{aligned}$$

To compute $\mathcal{E}[X(c)|\Gamma(c)=\gamma]$, we observe that algorithm p–Random may accept c only if it has rejected all calls of in its effective neighborhood γ. The probability that all calls of γ are rejected given that $\Gamma(c)=\gamma$ is $(1-p)^{|\gamma|}$, while then c is accepted with probability p. Thus,

$$\mathcal{E}[X(c)|\Gamma(c)=\gamma] = p(1-p)^{|\gamma|}. \quad (3.8)$$

Bounding from below $\mathcal{E}\left[\sum_{c'\in\gamma}\frac{X(c')}{d(c')}|\Gamma(c)=\gamma\right]$ is more complicated. In Sections 3.3.1 and 3.3.2, we consider two cases of sparse networks, i.e., networks of degree three and four.

Now, once we have computed a lower bound for $\mathcal{E}[Y(c)]$ for each call $c \in A(\sigma)$, by (2) and the definition of the competitive ratio, we can compute an upper bound on the competitive ratio $\rho(p)$ of algorithm p–Random as

$$\rho(p) \leq \left(\min_{c\in A(\sigma)}\mathcal{E}[Y(c)]\right)^{-1} \quad (3.9)$$

3.3.1 Analysis of algorithm p-Random in networks of maximum degree 3

Assume that the calls of the sequence σ appear in the cells of a network of degree three. Then, for each call c accepted by the optimal algorithm, we consider all possible configurations of the effective neighborhood of c. A configuration is depicted in Figure 3.8. The call c is the call accepted by the optimal algorithm. This call is adjacent to calls c_1, c_2, c_3. Arrows represent time; for example, the arrow from c_1 to c means that c_1 appeared prior to c. Note that a valid configuration has no directed cycle in the neighborhood of c and arrows between c and its adjacent calls are destined for c.

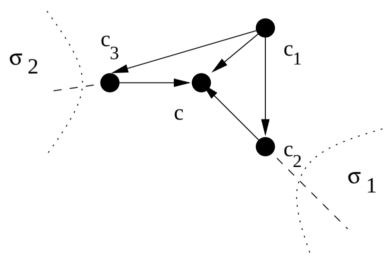

Figure 3.8: A configuration of the effective neighborhood of an optimal call c.

The sequence σ depicted in Figure 3.8 also contains the subsequences σ_1 and σ_2. The particular configuration for c will not be considered in our analysis for the following reason. Assume that the optimal algorithm accepts the set of calls $A(\sigma) = \{c\} \cup A(\sigma_1) \cup A(\sigma_2)$, where $A(\sigma_1)$ and $A(\sigma_2)$ denote the subsets of $A(\sigma)$ which consist of calls in σ_1 and σ_2, respectively. Note that $A'(\sigma) = \{c_3\} \cup A(\sigma_1) \cup A(\sigma_2)$ contains no mutually adjacent calls and, furthermore, has size equal to the size of $A(\sigma)$. Thus, we may assume that $A'(\sigma)$ is the set of optimal calls and consider a much simpler configuration for c_3. So, we obtain another constraint for the configurations we have to consider in our analysis: the optimal call has at most one common neighbor with any of its adjacent calls. In a network with degree three, the six configurations we have to consider are depicted in Figure 3.9. Symmetric cases have been omitted.

We denote by B_i the expectation of $Y(c)$ given that the effective neighborhood of c has the configuration γ_i, for $i = 0, 1, ..., 5$.

Clearly, by (3.4), we have that

$$B_0(c) \quad = \quad p \tag{3.10}$$

For configuration γ_1, we have that $d(c_1) \leq 3$, since the network has maximum degree three. Clearly, $\mathcal{E}\left[\sum_{c' \in \gamma} \frac{X(c')}{d(c')} | \Gamma(c) = \gamma_1\right] \geq p/3$ and, using

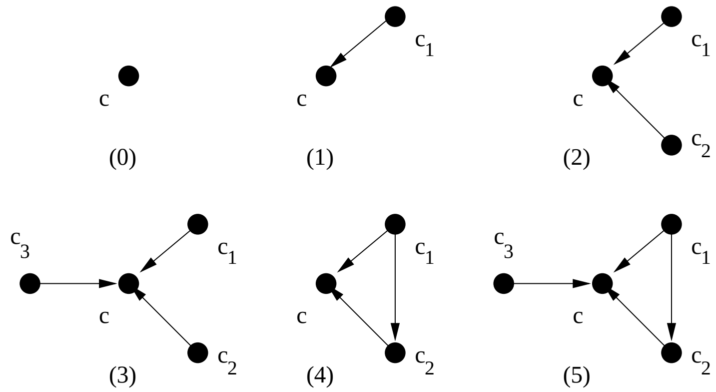

Figure 3.9: The six configurations we have to consider in our analysis. (i) corresponds to configuration γ_i.

(3.4), we obtain that

$$\begin{aligned} B_1(c) &\geq p(1-p) + p/3 \\ &= 4p/3 - p^2 \end{aligned} \tag{3.11}$$

For configuration γ_2, we again have that $d(c_1) \leq 3$ and $d(c_2) \leq 3$ and $\mathcal{E}\left[\sum_{c' \in \gamma} \frac{X(c')}{d(c')} | \Gamma(c) = \gamma_2\right] \geq 2p/3$. Using (3.4), we obtain that

$$\begin{aligned} B_2(c) &\geq p(1-p)^2 + 2p/3 \\ &= 5p/3 - 2p^2 + p^3 \end{aligned} \tag{3.12}$$

For configuration γ_3, we again have that $d(c_1) \leq 3$, $d(c_2) \leq 3$, $d(c_3) \leq 3$ and $\mathcal{E}\left[\sum_{c' \in \gamma} \frac{X(c')}{d(c')} | \Gamma(c) = \gamma_3\right] \geq p$. Using (3.4), we obtain that

$$\begin{aligned} B_3(c) &\geq p(1-p)^3 + p \\ &= 2p - 3p^2 + 3p^3 - p^4 \end{aligned} \tag{3.13}$$

For configuration γ_4, we have that $d(c_1) \leq 2$ and $d(c_2) \leq 2$, since the network has maximum degree three. To compute $\mathcal{E}\left[\sum_{c' \in \gamma} \frac{X(c')}{d(c')} | \Gamma(c) = \gamma_4\right]$ observe that c_1 is accepted with probability p while c_2 is accepted with probability p if c_1 was previously rejected, i.e., the probability that c_2 is accepted is $p(1-p)$. Thus,

$$\begin{aligned} \mathcal{E}\left[\sum_{c' \in \gamma} \frac{X(c')}{d(c')} | \Gamma(c) = \gamma_4\right] &\geq p/2 + p(1-p)/2 \\ &= p - p^2/2 \end{aligned}$$

Using (3.4), we obtain that

$$\begin{aligned} B_4(c) &\geq p(1-p)^2 + p - p^2/2 \\ &= 2p - 5p^2/2 + p^3 \end{aligned} \tag{3.14}$$

For configuration γ_5, we have that $d(c_1) \leq 2$, $d(c_2) \leq 2$ and $d(c_3) \leq 3$, since the network has maximum degree three. Using the same reasoning as above, we have

$$\begin{aligned} \mathcal{E}\left[\sum_{c' \in \gamma} \frac{X(c')}{d(c')} | \Gamma(c) = \gamma_5\right] &\geq p/2 + p(1-p)/2 + p/3 \\ &= 4p/3 - p^2/2 \end{aligned}$$

Using (3.4), we obtain that

$$\begin{aligned} B_4(c) &\geq p(1-p)^3 + 4p/3 - p^2/2 \\ &= 7p/3 - 7p^2/2 + 3p^3 - p^4 \end{aligned} \tag{3.15}$$

Now, the expectation of $Y(c)$ can be expressed as

$$\mathcal{E}[Y(c)] \geq \min_i B_i(c) \tag{3.16}$$

By making simple calculations with (3.7)–(3.11), we may verify that (since $p \in [0,1]$)

$$B_1(c) \leq \min\{B_2(c), B_3(c), B_4(c), B_5(c)\}.$$

For $p \geq 1/3$ (which contains the range of p in which we are interested for improving known results), (3.6) and (3.7) also yield $B_1(c) \leq B_0(c)$. Thus, for $p \in [1/3, 1]$, $B_1(c)$ is a lower bound for $\mathcal{E}[Y(c)]$ for each call $c \in A(\sigma)$ which only depends on p. Using (3.5), we obtain that for $p \in [1/3, 1]$, the competitive ratio of algorithm p–Random is

$$\rho(p) \leq \frac{3}{4p - 3p^2}.$$

The right side of the above inequality is minimized for $p = 2/3$ to $9/4$. We have obtained the following theorem.

Theorem 13. *There exists a $9/4$–competitive randomized on–line call control algorithm for sparse wireless cellular networks of degree three that support one frequency.*

Using the technique of Awerbuch et al., we can transform this algorithm to work with many frequencies. We obtain the following corollary.

Corollary 14. *There exists a* 2.787*–competitive randomized on–line call control algorithm for sparse wireless cellular networks of degree three that support many frequencies.*

3.3.2 Analysis of algorithm p-Random in networks of maximum degree 4

In this section we outline the proof for the competitiveness of algorithm p–Random on networks of degree four. Assume that the calls of the sequence σ appear in the cells of a network of degree four. Again, for each call c accepted by the optimal algorithm, we consider all possible configurations of the effective neighborhood of c. These configuration are constructed as in the previous section and now have the following constraints:

- At most four calls appear prior to c.
- There is no directed cycle in the neighborhood of c and arrows between c and its adjacent calls are destined for c.
- The optimal call c has at most two common neighbors with any of its adjacent calls.

In a network with degree four, we have to consider twenty three configurations instead of six that we considered for networks of degree three. Due to the limited space, we will not list all these configurations here. We denote by $B'_i(c)$ the expectation of $Y(c)$ given that the effective neighborhood of c has the configuration γ'_i, for $i = 0, 1, ..., 22$. We will just compute the expectation $B'_4(c)$ of $Y(c)$ given that the effective neighborhood of c has the configuration γ'_4 which is the same with configuration γ_4 of Figure 3.9.

We have that $d(c_1) \leq 3$ and $d(c_2) \leq 3$, since the network has maximum degree four. To compute $\mathcal{E}\left[\sum_{c'\in\gamma} \frac{X(c')}{d(c')}|\Gamma(c) = \gamma'_4\right]$, observe that c_1 is accepted with probability p while c_2 is accepted with probability p if c_1 was previously rejected, i.e., the probability that c_2 is accepted is $p(1-p)$. Thus,

$$\begin{aligned}\mathcal{E}\left[\sum_{c'\in\gamma} \frac{X(c')}{d(c')}|\Gamma(c) = \gamma_4\right] &\geq p/3 + p(1-p)/3 \\ &= 2p/3 - p^2/3\end{aligned}$$

Using (4), we obtain that

$$\begin{aligned} B'_4(c) &\geq p(1-p)^2 + 2p/3 - p^2/3 \\ &= 5p/3 - 7p^2/3 + p^3 \end{aligned} \tag{3.17}$$

Using similar reasoning as in Section 3.3.1, by examining all configurations, we can verify that, for some range of p, $B'_4(c) = \min_i B'_i(c)$. Thus, for this specific range of p, $B'_4(c)$ is a lower bound for $\mathcal{E}[Y(c)]$ for each call $c \in A(\sigma)$ which only depends on p. Using (3.5), we obtain that for the specific range of p, the competitive ratio $\rho'(p)$ of algorithm p–Random is

$$\rho'(p) \leq \frac{3}{5p - 7p^2 + 3p^3}.$$

The right side of the above inequality is minimized for $p = 5/9$ to $729/275 = 2.651$. We have obtained the following theorem.

Theorem 15. *There exists a* 2.651*–competitive randomized on–line call control algorithm for sparse wireless cellular networks of degree four that support one frequency.*

Surprisingly, this bound matches the best known result for ideal wireless cellular networks presented in [13]. Using the technique of Awerbuch et al., we can transform this algorithm to work with many frequencies. We obtain the following corollary.

Corollary 16. *There exists a* 3.182*–competitive randomized on–line call control algorithm for sparse wireless cellular networks of degree four that support many frequencies.*

3.4 Analysis of algorithm p-Random in arbitrary networks

Now, we extend the analysis of algorithm p–Random to arbitrary networks supporting one frequency. We express the result in terms of the maximum degree Δ of the network. Pantziou *et al.* [52] study the competitiveness of algorithm $\frac{1}{\Delta}$–Random on the average case, and prove that its competitive ratio on sequences of calls generated according to specific probability distributions is significantly smaller than the average degree of the network. Using a simple probabilistic argument, we prove the following result.

Theorem 17. *For any network with maximum degree $\Delta \geq 2$, there exists a $27\Delta/28$–competitive randomized call control algorithm against oblivious adversaries.*

Proof. Let G be a network with maximum degree Δ. Consider the execution of algorithm p–RANDOM (with $1/\Delta < p \leq 1$) on a sequence σ of calls on G. Following the notation of Theorem 7, we have that the expectation of the amortized benefit for each call c accepted by the optimal algorithm is

$$\mathcal{E}[\bar{b}(c)] \quad \geq \quad X(c) + \frac{\sum_{c' \in \gamma'(c)} X(c')}{\Delta},$$

while the expectation of the benefit of the algorithm on σ is given by

$$\mathcal{E}[B(\sigma)] \quad = \quad \sum_{c \in A(\sigma)} \mathcal{E}[\bar{b}(c)].$$

Let $q(c)$ be the probability that $\sum_{c' \in \gamma'(c)} X(c') = 0$, i.e., that no call is accepted in the effective neighborhood of c. Then

$$\begin{aligned} \mathcal{E}[\bar{b}(c)] &\geq q(c)p + \frac{1 - q(c)}{\Delta} \\ &= q(c)\left(p - \frac{1}{\Delta}\right) + \frac{1}{\Delta}. \end{aligned}$$

Note that $q(c) \geq (1-p)^{\Delta}$, for any $c \in \sigma$. Thus,

$$\mathcal{E}[\bar{b}(c)] \quad \geq \quad (1-p)^{\Delta}\left(p - \frac{1}{\Delta}\right) + \frac{1}{\Delta}.$$

The right part of the expression is maximized for $p = \frac{2}{\Delta+1}$. In this case, we obtain that

$$\begin{aligned} \mathcal{E}[\bar{b}(c)] &\geq \left(1 - \frac{2}{\Delta+1}\right)^{\Delta}\left(\frac{2}{\Delta+1} - \frac{1}{\Delta}\right) + \frac{1}{\Delta} \\ &= \frac{1}{\Delta}\left(\left(1 - \frac{2}{\Delta+1}\right)^{\Delta+1} + 1\right) \\ &\geq \frac{1}{\Delta}\left(\frac{1}{27} + 1\right) \\ &= \frac{28}{27\Delta} \end{aligned}$$

We conclude that[1], for the benefit of algorithm $\frac{2}{\Delta+1}$–RANDOM, the following inequality holds:

$$B_{OPT}(\sigma) \leq \frac{27\Delta}{28}\mathcal{E}[B(\sigma)].$$

The theorem follows. □

Note that using a theorem of Awerbuch *et al.* [1], theorem 17 implies the existence of a randomized call control algorithm for arbitrarily many frequencies with competitive ratio at most $27\Delta/28+1$.

Now consider a network G and let $\alpha(G)$ be the maximum independent set in the neighborhood of each node. As pointed out in Section 3.1, for any network, $|\alpha(G)|$ is a lower bound on the competitive ratio of any deterministic algorithm. By inserting $|\alpha(G)|$ in the proof of Theorem 17, we obtain the following.

Corollary 18. *For any network that supports one frequency with* $|\alpha(G)| \geq 2$ *and* $1/2 < p < 1$*, algorithm* p*–*RANDOM *has (strictly) better competitive ratio than any deterministic algorithm.*

Obviously, if $|\alpha(G)| = 1$ or $|\alpha(G)| = 0$, the greedy deterministic algorithm is optimal.

3.5 A simple 8/3-competitive algorithm

In this section we present algorithm CRS-A, a simple randomized online algorithm for call control in cellular networks of reuse distance 2. The algorithm works in networks with one frequency and achieves a competitive ratio against oblivious adversaries which is similar (but slightly inferior) to that which has been proved for algorithm p-RANDOM.

Algorithm CRS-A uses a coloring of the cells with four colors 0, 1, 2, and 3, such that only two colors are used in the cells belonging to the same axis. This can be done by coloring the cells in the same x-axis with either the colors 0 and 1 or the colors 2 and 3, coloring the cells in the same y-axis with either the colors 0 and 2 or the colors 1 and 3, and coloring the cells in

[1]Observe that the inequalities on $\mathcal{E}[\bar{b}(c)]$ imply that

$$\lim_{\Delta\to\infty} \Delta\mathcal{E}[\bar{b}(c)] = 1 + e^{-2}.$$

Thus, for graphs with sufficiently large maximum degree Δ, algorithm $\frac{2}{\Delta+1}$–RANDOM is at most $(0.8808+\epsilon)\Delta$–competitive for some small $\epsilon > 0$.

the same z-axis with either the colors 0 and 3 or the colors 1 and 2. Such a coloring is depicted in the left part of Figure 3.10.

Algorithm CRS-A randomly selects one out of the four colors and executes the greedy algorithm on the cells colored with the other three colors, ignoring (i.e., rejecting) all calls in cells colored with the selected color.

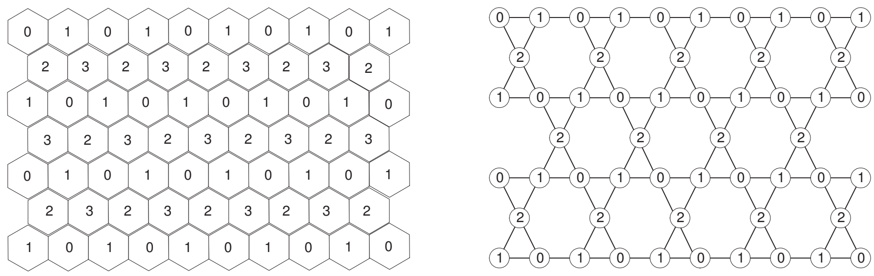

Figure 3.10: The 4-coloring used by algorithm CRS-A and the corresponding subgraph of the interference graph induced by the nodes not colored with color 3.

Theorem 19. *Algorithm CRS-A in cellular networks of reuse distance* 2 *supporting one frequency is* 8/3*-competitive against oblivious adversaries.*

Proof. Let σ be a sequence of calls and denote by O the set of calls accepted by the optimal algorithm. Denote by σ' the set of calls in cells which are not colored with the color selected and by O' the set of calls the optimal algorithm would have accepted on input σ'. Clearly, $|O'|$ will be at least as large as the subset of O which belongs to σ'. Since the probability that the cell of a call in O is not colored with the color selected is $3/4$, it is $\mathcal{E}[|O'|] \geq \frac{3}{4}|O|$. Now let B be the set of calls accepted by algorithm CRS-A, i.e., the set of calls accepted by the greedy algorithm when executed on sequence σ'. Observe that each call in O' either belongs in B or it is rejected because some other call is accepted. Furthermore, a call in $B \setminus O'$ can cause the rejection of at most two calls of O'. This implies that $|B| \geq |O'|/2$ which yields that the competitive ratio of algorithm CRS-A is

$$\frac{|O|}{\mathcal{E}[|B|]} \leq \frac{2|O|}{\mathcal{E}[|O'|]} \leq \frac{8}{3}.$$

□

The main advantage of algorithm CRS-A is that it uses only two random bits. In the next section we present simple on-line algorithms with improved competitive ratios that use slightly stronger random sources and work on networks with arbitrarily many frequencies.

3.6 *““Classify and Randomly select””*-based Algorithms

Algorithm CRS-A can be seen as an algorithm based on the “classify and randomly select” paradigm. It uses a coloring of the interference graph (not necessarily using the minimum possible number of colors) and a classification of the colors. It starts by randomly selecting a color class (i.e., a set of colors) and then run the greedy algorithm in the cells colored with colors from this color class, ignoring (i.e., rejecting) calls in cells colored with colors not belonging to this class. Algorithm CRS-A uses a coloring of the interference graph with four colors 0, 1, 2, and 3, and the four color classes $\{0, 1, 2\}, \{0, 1, 3\}, \{0, 2, 3\}$, and $\{1, 2, 3\}$. Note that, in the previously known algorithms based on the “classify and randomly select” paradigm, color classes are singletons (e.g., [2], [52]).

The following simple lemma gives a sufficient condition for obtaining efficient on-line algorithms based on the “classify and randomly select” paradigm.

Lemma 20. *Consider a network with interference graph $G = (V, E)$ which supports w frequencies and let χ be a coloring of the nodes of V with the colors of a set X. If there exist ν sets of colors $s_0, s_1, ..., s_{\nu-1} \subseteq X$ and an integer λ such that*

- *each color of X belongs to at least λ different sets of the sets $s_0, s_1, ..., s_{\nu-1}$, and*
- *for $i = 0, 1, ..., \nu - 1$, each connected component of the subgraph of G induced by the nodes colored with colors in s_i is a clique,*

then there exists an on-line randomized call control algorithm for the network G which has competitive ratio ν/λ against oblivious adversaries.

Proof. Consider a network with interference graph G which supports w frequencies and the randomized on-line algorithm working as follows. The algorithm randomly selects one out of the ν color classes $s_0, ..., s_{\nu-1}$ and executes the greedy algorithm on the cells colored with colors of the selected class, rejecting all calls in cells colored with colors not in the selected class.

Let σ be a sequence of calls and let O be the set of calls accepted by the optimal algorithm on input σ. Assume that the algorithm selects the color class s_i. Let σ' be the sequence of calls in cells colored with colors in s_i and O' be the set of calls accepted by the optimal algorithm on input σ'. Also, we denote by B the set of calls accepted by the algorithm. First we can easily show that $|B| = |O'|$. Let G_j be a connected component of the subgraph of G induced by the nodes of G colored with colors in s_i. Let σ_j be the subsequence of σ' in cells corresponding to nodes of G_j. Clearly, any algorithm (including the optimal one) will accept at most one call of σ_j at each frequency. If the optimal algorithm accepts w calls, this means that the sequence σ_j has at least w calls and the greedy algorithm, when executed on σ', will accept w calls from σ_j (one call in each one of the available frequencies). If the optimal algorithm accepts $w' < w$ calls from σ_j, this means that σ_j contains exactly $w' < w$ calls and the greedy algorithm will accept them all in w' different frequencies. Since a call of σ_j is not constrained by a call in $\sigma_{j'}$ for $j \neq j'$, we obtain that $|B| = |O'|$.

The proof is completed by observing that the expected benefit of the optimal algorithm on input σ' over all possible sequences σ' defined by the random selection of the algorithm is $\mathcal{E}[|O'|] \geq \frac{\nu}{\lambda}|O|$, since, for each call in O, the probability that the color of its cell belongs to the color class selected is at least ν/λ. Hence, the competitive ratio of the algorithm against oblivious adversaries is

$$\frac{|O|}{\mathcal{E}[|B|]} = \frac{|O|}{\mathcal{E}[|O'|]} \leq \lambda/\nu.$$

□

Next, we present two randomized on-line algorithms for call control in cellular networks of reuse distance 2, namely CRS-B and CRS-C, which are also based on the "classify and randomly select" paradigm and achieve even better competitive ratios.

Consider a coloring of the cells with five colors $0, 1, 2, 3$, and 4 such that for each $i \in \{0, 1, 2, 3, 4\}$, and for each cell colored with color i, the two adjacent cells in the same x-axis are colored with colors $(i-1) \bmod 5$ and $(i+1) \bmod 5$, while the remaining four of its adjacent cells are colored with colors $(i+2) \bmod 5$ and $(i+3) \bmod 5$. Such a coloring is depicted in the left part of Figure 3.11. Also, define $s_i = \{i, (i+1) \bmod 5\}$, for $i = 0, 1, ..., 4$. Observe that, for each $i = 0, 1, ..., 4$, each pair of adjacent cells colored with the colors i and $(i+1) \bmod 5$ is adjacent to cells colored with colors $(i+2) \bmod 5$, $(i+3) \bmod 5$, and $(i+4) \bmod 5$, i.e., colors not belonging to s_i. Thus, the coloring χ together with the color classes s_i satisfy the conditions of Lemma 20 with $\nu = 5$ and $\lambda = 2$. We call CRS-B the algorithm that uses this coloring and works according to the "classify and randomly select" paradigm as in the proof of Lemma 20. We obtain the following.

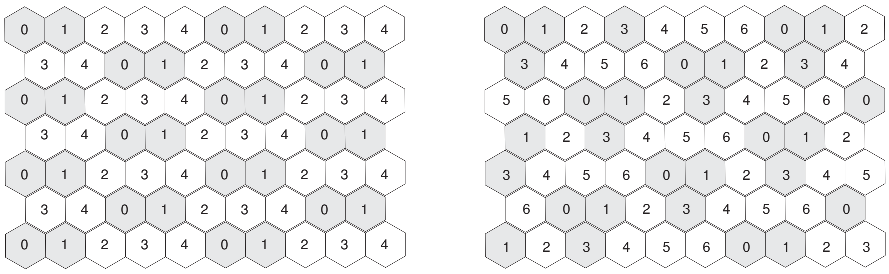

Figure 3.11: The 5-coloring used by algorithm CRS-B and the 7-coloring used by algorithm CRS-C. The skewed cells are those colored with the colors in set s_0.

Theorem 21. *Algorithm CRS-B in cellular networks with reuse distance* 2 *is* 5/2*-competitive against oblivious adversaries.*

Now consider a coloring of the cells with seven colors $0, 1, ..., 6$ such that for each for each cell colored with color i (for $i = 0, ..., 6$), its two adjacent cells in the same x-axis are colored with the colors $(i-1) \bmod 7$ and $(i+1) \bmod 7$, while its two adjacent cells in the same z-axis are colored with colors $(i-3) \bmod 7$ and $(i+3) \bmod 7$. Such a coloring is depicted in the right part of Figure 3.11. Also, define $s_i = \{i, (i+1) \bmod 7, (i+3) \bmod 7\}$, for $i = 0, 1, ..., 6$. Observe that, for each $i = 0, 1, ..., 6$, each triangle of cells colored with the colors i, $(i+1) \bmod 7$, and $(i+3) \bmod 7$ is adjacent to cells colored with colors $(i+2) \bmod 7$, $(i+4) \bmod 7$, $(i+5) \bmod 7$, and $(i+6) \bmod 7$, i.e., colors not belonging to s_i. Thus, the coloring χ together with the color classes s_i satisfy the conditions of Lemma 20 with $\nu = 7$ and $\lambda = 3$. We call CRS-C the algorithm that uses this coloring and works according to the "classify and randomly select" paradigm as in the proof of Lemma 20. We obtain the following.

Theorem 22. *Algorithm CRS-C in cellular networks with reuse distance* 2 *is* 7/3*-competitive against oblivious adversaries.*

The two algorithms above (CRS-B and CRS-C) make use of a random source which equiprobably selects one out of an odd number of distinct objects. If we only have a number of fair coins (random bits) available, we can design algorithms with small competitive ratios by combining the algorithms above. For example, using 6 random bits, we may construct the following algorithm. We use integers $0, 1, ..., 63$ to identify each of the 63 outcomes of the 6 random bits. For an outcome $i \in \{0, ..., 49\}$, the algorithm executes algorithm CRS-B using color class $s_{i \bmod 5}$, and for an outcome $i \in \{50, ..., 63\}$, the algorithm executes algorithm CRS-C using color class $s_{(i-50) \bmod 7}$. It can be easily seen that this algorithm has competitive ratio $32/13 \approx 2.462$ against oblivious adversaries, since its expected benefit is at least $50/64 \cdot 2/5 + 14/64 \cdot 3/7 = 13/32$ times the optimal benefit. Similarly, using 8 random bits, we obtain an on-line algorithm with competitive ratio $64/27 \approx 2.37$. We can generalize this idea, and, for sufficiently small $\epsilon > 0$, we can construct an algorithm which uses $t = O(\log 1/\epsilon)$ random bits, and, on $2^t \bmod 7$ of the 2^t outcomes, it does nothing, while the rest of the outcomes are assigned to executions of algorithm CRS-C. In this way, we obtain the following.

Corollary 23. *For any $\epsilon > 0$, there exists an on-line randomized call-control algorithm for cellular networks with reuse distance 2 that uses $O(\log 1/\epsilon)$ random bits and has competitive ratio at most $7/3 + \epsilon$ against oblivious adversaries.*

3.7 Networks of reuse distance $k > 2$

For cellular networks with reuse distance $k > 2$, we present algorithm CRS-k which is based on the "classify and randomly select" paradigm. Algorithm CRS-k uses the following coloring of the interference graph of a cellular network with reuse distance k. Cells are colored with the colors $0, 1, ..., 3k^2 - 3k$ such that for any cell colored with color i, its adjacent cells in the x-axis are colored with colors $(i-1) \bmod (3k^2-3k+1)$ and $(i+1) \bmod (3k^2-3k+1)$, while its adjacent cells in the z-axis are colored with colors $(i-3(k-1)^2) \bmod (3k^2-3k+1)$ and $(i+3(k-1)^2) \bmod (3k^2-3k+1)$.

For odd k, for $i = 0, 1, ..., 3k^2-3k$, the color class s_i contains the following colors. For $j = 0, 1, ..., \frac{k-1}{2}$, it contains the colors $(i + 3j(k-1)^2 - j) \bmod (3k^2-3k+1)$, ..., $(i + 3j(k-1)^2 + \frac{k-1}{2}) \bmod (3k^2-3k+1)$, and for $j = \frac{k+1}{2}, ..., k-1$, it contains the colors $(i + \frac{3(k-1)^3}{2} + 3(j - \frac{k-1}{2})(k-1)^2 - \frac{k-1}{2}) \bmod (3k^2-3k+1)$, ..., $(i + \frac{3(k-1)^3}{2} + 3(j - \frac{k-1}{2})(k-1)^2 + k - 1 - j) \bmod (3k^2-3k+1)$.

For even k, for $i = 0, 1, ..., 3k^2-3k$, the color class s_i contains the following colors. For $j = 0, 1, ..., \frac{k}{2} - 1$, it contains the colors $(i + 3j(k-1)^2 - j) \bmod (3k^2-3k+1), ..., (i + 3j(k-1)^2 + \frac{k}{2}) \bmod (3k^2-3k+1)$, and for $j = \frac{k}{2}, ..., k-1$, it contains the colors $(i + 3(\frac{k}{2} - 1)(k-1)^2 + 3(j - \frac{k}{2} + 1)(k-1)^2 - \frac{k}{2} + 1) \bmod (3k^2 - 3k + 1)$, ..., $(i + 3(\frac{k}{2} - 1)(k-1)^2 + 3(j - \frac{k}{2} + 1)(k-1)^2 + k - 1 - j) \bmod (3k^2 - 3k + 1)$.

Note that, for $k = 2$, we obtain the coloring used by algorithm CRS-C. Examples of the coloring for $k = 3$ and $k = 4$ as well as the cells colored with colors from the color class s_0 are depicted in Figure 3.12.

We can show the following two lemmas.

Lemma 24. *Let $k > 2$ and G be the interference graph of a cellular network of distance reuse k. Consider the coloring of G used by algorithm CRS-k and the color classes s_i, for $i = 0, 1, ..., 3k^2 - 3k$. For any color j such that $0 \leq j \leq 3k^2 - 3k$, the number of different color classes s_i that color j belongs to is $\frac{3k^2}{4}$ if k is even, and $\frac{3k^2+1}{4}$ if k is odd.*

Lemma 25. *Let $k > 2$ and G be the interference graph of a cellular network of distance reuse k. Consider the coloring of G used by algorithm CRS-k and the color classes s_i, for $i = 0, 1, ..., 3k^2 - 3k$. For $i = 0, 1, ..., 3k^2 - 3k$, each connected component of the subgraph of G induced by the nodes of G colored with colors in s_i is a clique.*

Thus, the colorings and the color classes described satisfy the condition of Lemma 20 with $\lambda = 3k^2 - 3k + 1$ and $\nu = \frac{3k^2}{4}$ if k is even, and $\nu = \frac{3k^2+1}{4}$ if k is odd. By Lemma 20, we obtain the following.

Theorem 26. *The competitive ratio against oblivious adversaries of algorithm CRS-k in cellular networks with reuse distance $k \geq 2$ is $4\left(1 - \frac{3k-1}{3k^2}\right)$ if k is even, and $4\left(1 - \frac{3k}{3k^2+1}\right)$ if k is odd.*

Algorithm CRS-k in cellular networks of reuse distance k uses a random source which equiprobably selects one among $3k^2 - 3k + 1$ distinct objects. By applying ideas we used in the previous section, we can achieve similar competitiveness bounds by algorithms that use random bits.

Corollary 27. *For any $\epsilon > 0$, there exists an on-line randomized call-control algorithm for cellular networks of reuse distance k, that uses $O(\log 1/\epsilon + \log k)$ random bits and has competitive ratio at most $4\left(1 - \frac{3k-1}{3k^2}\right) + \epsilon$ if k is even, and $4\left(1 - \frac{3k}{3k^2+1}\right) + \epsilon$ if k is odd, against oblivious adversaries.*

3.8 Lower Bounds

In Section 3.1, it has been observed that no deterministic on-line call control algorithm can have a competitive ratio better than 3 against off-line adversaries. We can easily extend this lower bound and obtain lower

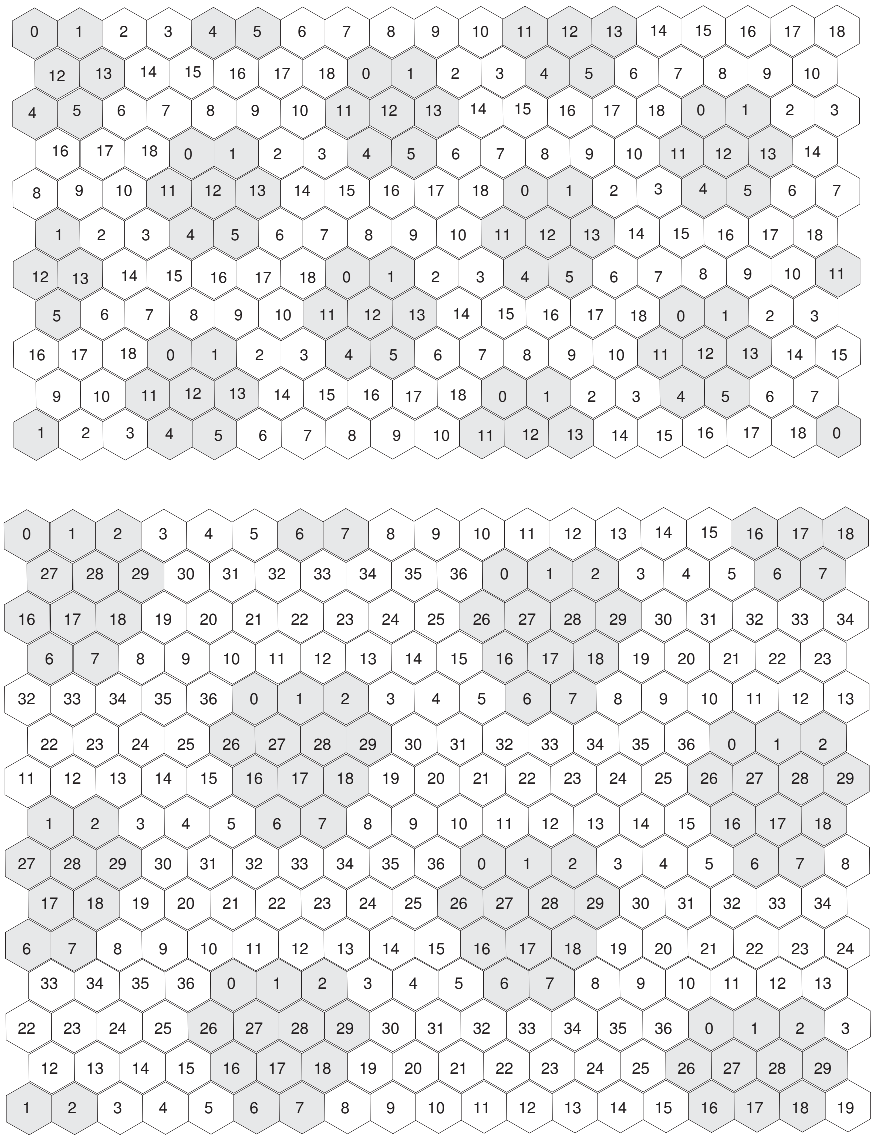

Figure 3.12: Examples of the colorings used by the algorithms CRS-3 and CRS-4. The skewed cells are those colored with the colors in set s_0.

bounds of 4 and 5 on the competitiveness of deterministic on-line call control algorithms in cellular networks of reuse distance $k \in \{3, 4, 5\}$ and $k \geq 6$, respectively.

Consider a cellular network of reuse distance k supporting one frequency. The lower bound can be easily extended for networks supporting arbitrarily many frequencies. Let v be a cell of the network and consider the cells at distance at most $k-1$ from v. It can be easily verified that we can pick a set S of $\left\lfloor \frac{6(k-1)}{k} \right\rfloor$ cells at distance at most $k-1$ from v such that the distance between any two of them is at least k. Consider a deterministic algorithm A and an off-line adversary $\mathcal{ADV}$ working as follows. First the adversary presents a call in cell v. If the algorithm A rejects the call, the adversary stops the sequence; in this case A has no benefit. Otherwise (i.e., if A accepts the call in cell v), the adversary presents one call in each of the cells of S. In this case, the benefit of the algorithm A is 1, while the optimal off-line algorithm has benefit the size of S which is 4 if $k \in \{3, 4, 5\}$ and 5 if $k \geq 6$.

Hence, the randomized algorithms presented in the previous section significantly beat the lower bound on the competitiveness of deterministic algorithms. In what follows, using the Minimax Principle [61] (see also [47]), we prove a lower bound on the competitive ratio, against oblivious adversaries, of any randomized algorithm in cellular networks with reuse distance $k \in \{2, 3, 4\}, k \in \{5, \cdots, 11\}, k \geq 12$, as well as for cellular networks with planar interference graphs. Again, we consider networks that support one frequency; our lower bound can be trivially extended to networks that support multiple frequencies.

Lemma 28 (Minimax Principle [47]). *Given a probability distribution $\mathcal{P}$ over sequences of calls σ, denote by $\mathcal{E}_{\mathcal{P}}[B_A(\sigma)]$ and $\mathcal{E}_{\mathcal{P}}[B_{OPT}(\sigma)]$ the expected benefit of a deterministic algorithm A and the optimal off–line algorithm on sequences of calls generated according to $\mathcal{P}$. Define the competitiveness of A under $\mathcal{P}$, $c_A^{\mathcal{P}}$ to be such that*

$$c_A^{\mathcal{P}} = \frac{\mathcal{E}_{\mathcal{P}}[B_{OPT}(\sigma)]}{\mathcal{E}_{\mathcal{P}}[B_A(\sigma)]}.$$

Let A_R be a randomized algorithm. Then, the competitiveness of A under $\mathcal{P}$ is a lower bound on the competitive ratio of A_R against an oblivious adversary, i.e. $c_A^{\mathcal{P}} \leq c_{A_R}$.

Theorem 29. *No randomized call–control algorithm can be better than 1.857–competitive against an oblivious adversary when applied to cellular networks of reuse distance $k \in \{2, 3, 4\}$.*

Proof. Consider a proper coloring of the cells of the network with the colors RED, BLUE, and GREEN. Let r_0 be a red cell, b_1, b_2, and b_3 its blue neighbors, and g_1, g_2, and g_3 its green neighbors. We will prove that there exists an adversary $\mathcal{ADV}$ that produces calls according to a probability distribution $\mathcal{P}$ in such way that no deterministic algorithm can be better than 1.857–competitive under $\mathcal{P}$ even if it knows the probability distribution $\mathcal{P}$ in advance.

We define the probability distribution $\mathcal{P}$ as follows. First, the adversary produces a call in the red cell r_0. Then, it

- either stops, with probability 4/7,
- or does the following, with probability 3/7. It presents two calls in the cells b_1 and b_2, and
  - either produces a call in the cell b_3 with probability 1/3,
  - or presents three calls in the cells g_1, g_2, and g_3, with probability 2/3.

It can be easily seen that the expected benefit of the optimal off–line algorithm on sequences of calls generated according to $\mathcal{P}$ is

$$\mathcal{E}_{\mathcal{P}}[B_{OPT}(\sigma)] = 1 \cdot \frac{4}{7} + 3 \cdot \frac{3}{7} = \frac{13}{7}.$$

Let A be a deterministic call control algorithm that runs on the calls produced by $\mathcal{ADV}$. Consider t executions of the algorithm on t sequences produced according to the probability distribution $\mathcal{P}$. Let q_0 be the number of executions in which A accepts the call produced in cell r_0, and q_1 the number of executions in which A accepts both calls in cells b_1 and b_2.

The expected number of executions in which the algorithm does not accept the call in cell r_0 and the adversary produces a call in cell b_3 is $\frac{3}{7}\frac{1}{3}(t-q_0)$. Similarly, the expected number of executions in which the algorithm does not accept the calls in cells r_0, b_1, and b_2 and the adversary produces calls in cells g_1, g_2, g_3 is $\frac{2}{3}\left(\frac{3}{7}(t-q_0)-q_1\right)$. Thus,

$$\begin{aligned}\mathcal{E}_{\mathcal{P}}[B_A(\sigma)] &\leq \frac{q_0 + 2q_1 + \frac{3}{7}\frac{1}{3}(t-q_0) + 3\frac{2}{3}\left(\frac{3}{7}(t-q_0)-q_1\right)}{t} \\ &= 1\end{aligned}$$

and

$$c_A^{\mathcal{P}} \geq \frac{13}{7} = 1.857.$$

By Lemma 28, we obtain that this is a lower bound on the competitive ratio of any randomized algorithm against an oblivious adversary. □

Theorem 30. *No randomized call–control algorithm in cellular networks with distance reuse $k \geq 5$ can be better than $25/12$–competitive against an oblivious adversary.*

Proof. Consider a cellular network with reuse distance 5 and ten cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$ as shown in Figure 3.14. We will prove that there exists an adversary $\mathcal{ADV}$ that produces calls in these cells according to a probability distribution $\mathcal{P}$ in such way that no deterministic algorithm can be better than 25/12–competitive under $\mathcal{P}$ even if it knows the probability distribution $\mathcal{P}$ in advance.

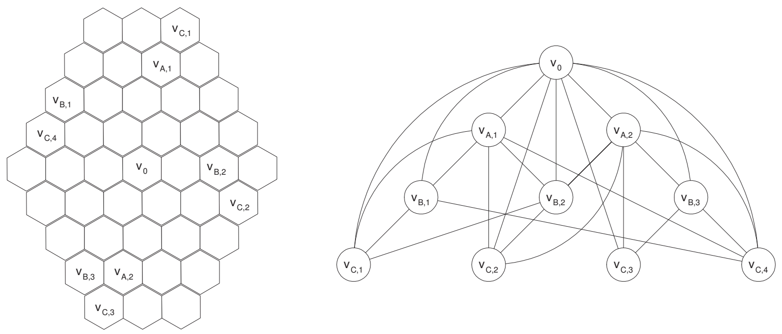

Figure 3.13: The cellular network of reuse distance 5 used in the proof of Theorem 9 and the subgraph of the interference graph induced by the nodes corresponding to the ten cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$.

We define the probability distribution $\mathcal{P}$ as follows. First, the adversary produces a call in the cell v_0. Then, it

- either stops, with probability 1/2,
- or does the following, with probability 1/2. It presents two calls, one in the cell $v_{A,1}$ and one in the cell $v_{A,2}$, and
  - either stops, with probability 1/3,
  - or does the following with probability 2/3. It presents three calls, one in the cell $v_{B,1}$, one in the cell $v_{B,2}$, and one in the cell $v_{B,3}$, and

* either stops, with probability 1/4,
* or does the following, with probability 3/4. It presents four calls, one in the cell $v_{C,1}$, one in the cell $v_{C,2}$, one in the cell $v_{C,3}$, and one in the cell $v_{C,4}$, and then stops.

Clearly, the benefit of the optimal off-line algorithm is the number of calls presented by the adversary in the last step before stopping the sequence. Thus, the expected benefit of the optimal off–line algorithm on sequences of calls generated according to $\mathcal{P}$ is

$$\mathcal{E}_{\mathcal{P}}[B_{OPT}(\sigma)] = 1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{2} \cdot \frac{1}{3} + 3 \cdot \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{1}{4} + 4 \cdot \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{3}{4} = \frac{25}{12}.$$

Let A be a deterministic call control algorithm that runs on the calls produced by $\mathcal{ADV}$. First, we observe that no algorithm could gain by accepting one of the two calls presented in cells $v_{A,1}$ and $v_{A,2}$ or by accepting one or two of the three calls presented in cells $v_{B,1}$, $v_{B,2}$ and $v_{B,3}$. Assume that it rejects the call in cell v_0, accepts the call in cell $v_{A,1}$ (resp. $v_{A,2}$) and rejects the call in cell $v_{A,2}$ (resp. $v_{A,1}$). Then, even if the adversary does not stop the sequence just after producing the two calls in cells $v_{A,1}$ and $v_{A,2}$, the algorithm can only accept one of the two calls in the cells $v_{B,3}$ and $v_{C,3}$ (resp. $v_{B,1}$ and $v_{C,1}$); this would give a benefit of at most 2. Also, no algorithm could gain by accepting one of the three calls presented in cells $v_{B,1}$, $v_{B,2}$, and $v_{B,3}$. Assume that the algorithm rejects the calls in cells v_0, $v_{A,1}$ and $v_{A,2}$, accepts the call in cell $v_{B,1}$ (resp. $v_{B,2}$, resp. $v_{B,3}$), and rejects the calls in cells $v_{B,2}$ and $v_{B,3}$ (resp. $v_{B,1}$ and $v_{B,3}$, resp. $v_{B,2}$ and $v_{B,3}$). Then, even if the adversary does not stop the sequence before producing the calls in the cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$, the algorithm can only accept the calls in the cells $v_{C,2}$ and $v_{C,3}$ (resp. $v_{C,3}$ and $v_{C,4}$, resp. $v_{C,1}$ and $v_{C,2}$); this would give a benefit of at most 3. Also, no algorithm could gain by accepting two of the three calls presented in cells $v_{B,1}$, $v_{B,2}$, and $v_{B,3}$. Assume that the algorithm rejects the calls in the cells v_0, $v_{A,1}$, and $v_{A,2}$, accepts the calls in the cells $v_{B,2}$ and $v_{B,3}$ and rejects the call in cell $v_{B,1}$. Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$, the algorithm cannot accept any more calls. Similarly, assume that the algorithm rejects the calls in cells v_0, $v_{A,1}$ and $v_{A,2}$, accepts the calls in cells $v_{B,1}$ and $v_{B,2}$ (resp. $v_{B,1}$ and $v_{B,3}$), and rejects the call in cell $v_{B,3}$ (resp. $v_{B,2}$). Then, even if the adversary does not stop the sequence before producing the calls in the cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$, the best the algorithm can do is to accept the call in cell $v_{C,3}$ (resp. $v_{C,2}$); this would give a benefit of at most 3. The above observations can be easily made by studying carefully the subgraph of the interference graph induced by the ten cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$ (see right part of Figure 3.14). Hence, we may assume that the algorithm either accepts all calls presented at a step or rejects them all.

Consider t executions of the algorithm on t sequences produced according to the probability distribution $\mathcal{P}$. Let q_0 be the number of executions in which A accepts the call produced in cell v_0, q_1 the number of executions in which A accepts both calls in cells $v_{A,1}$ and $v_{A,2}$, and q_2 the number of executions in which the algorithm accepts the three calls in cells $v_{C,1}$, $v_{C,2}$, and $v_{C,3}$.

The expected number of executions in which the algorithm does not accept the call in cell v_0 and the adversary produces calls in cells $v_{A,1}$ and $v_{A,2}$ is $\frac{1}{2}(t-q_0)$. Hence, the expected number of executions in which the algorithm does not accept the calls in cells v_0, $v_{A,1}$, and $v_{A,2}$ and the adversary produces calls in cells $v_{B,1}$, $v_{B,2}$, and $v_{B,3}$ is $\frac{2}{3}\left(\frac{1}{2}(t-q_0)-q_1\right)$ and the expected number of executions in which the algorithm does not accept the calls in cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{B,1}$, $v_{B,2}$, and $v_{B,3}$ and the adversary produces calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$ is $\frac{3}{4}\left(\frac{2}{3}\left(\frac{1}{2}(t-q_0)-q_1\right)-q_2\right)$. Thus,

$$\mathcal{E}_{\mathcal{P}}[B_A(\sigma)] \leq \frac{q_0+2q_1+3q_2+4\frac{3}{4}\left(\frac{2}{3}\left(\frac{1}{2}(t-q_0)-q_1\right)-q_2\right)}{t} = 1$$

and $c_A^{\mathcal{P}} \geq 25/12$. By Lemma 28, we obtain that this is a lower bound on the competitiveness of any randomized algorithm against oblivious adversaries. □

Theorem 31. *No randomized call–control algorithm in cellular networks with distance reuse $k=12$ can be better than $127/60$–competitive against an oblivious adversary.*

Proof. Consider a cellular network with distance reuse 12 and thirteen cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{A,3}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{B,4}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$ and $v_{C,5}$ as shown in Figure 3.14. We will prove that there exists an adversary $\mathcal{ADV}$ that produces calls in these cells according to a probability distribution $\mathcal{P}$ in such way that no deterministic algorithm can be better than 127/60–competitive under $\mathcal{P}$ even if it knows the probability distribution $\mathcal{P}$ in advance.

We define the probability distribution $\mathcal{P}$ as follows. First, the adversary produces a call in the cell v_0. Then, it

- either stops, with probability 2/3,
- or does the following, with probability 1/3. It presents three calls, one in the cell $v_{A,1}$, one in the cell $v_{A,2}$ and one in the cell $v_{A,3}$, and

Figure 3.14: The cellular network of distance reuse 12 used in the proof of Theorem 9 and the subgraph of the interference graph induced by the nodes corresponding to the thirteen cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{A,3}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{B,4}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$ and $v_{C,5}$.

- either stops, with probability 1/4,
- or does the following with probability 3/4. It presents four calls, one in the cell $v_{B,1}$, one in the cell $v_{B,2}$, one in the cell $v_{B,3}$, and one in the cell $v_{B,4}$, and
  * either stops, with probability 1/5,
  * or does the following, with probability 4/5. It presents five calls, one in the cell $v_{C,1}$, one in the cell $v_{C,2}$, one in the cell $v_{C,3}$, one in the cell $v_{C,4}$, and one in the cell $v_{C,5}$, and then stops.

Clearly, the benefit of the optimal off-line algorithm is the number of calls presented by the adversary in the last step before stopping the sequence. Thus, the expected benefit of the optimal off–line algorithm on sequences of calls generated according to $\mathcal{P}$ is

$$\mathcal{E}_{\mathcal{P}}[B_{OPT}(\sigma)] = 1 \cdot \frac{2}{3} + 3 \cdot \frac{1}{3} \cdot \frac{1}{4} + 4 \cdot \frac{1}{3} \cdot \frac{3}{4} \cdot \frac{1}{5} + 5 \cdot \frac{1}{3} \cdot \frac{3}{4} \cdot \frac{4}{5} = \frac{127}{60}.$$

Let A be a deterministic call control algorithm that runs on the calls produced by $\mathcal{ADV}$. First, we observe that no algorithm could gain by accepting one of the three calls presented in cells $v_{A,1}$, $v_{A,2}$ and $v_{A,3}$. Assume that it rejects the call in cell v_0, accepts the call in cell $v_{A,1}$ (resp. $v_{A,2}$, resp.$v_{A,3}$) and rejects the call in cell $v_{A,2}$ and $v_{A,3}$ (resp. $v_{A,1}$ and $v_{A,2}$, resp. $v_{A,1}$ and $v_{A,2}$). Then, even if the adversary does not stop the sequence just after producing the three calls in cells $v_{A,1}$, $v_{A,2}$ and $v_{A,3}$, the algorithm can only accept the two calls in the cells $v_{B,3}$ and $v_{B,4}$ (resp. $v_{B,1}$ and $v_{B,4}$, resp. $v_{B,1}$ and $v_{B,2}$); this would give a benefit of at most 3.

Similarly, no algorithm could gain by accepting two of the three calls presented in cells $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$. Assume that the algorithm rejects the calls in the cells v_0, accepts the calls in the cells $v_{A,1}$ and $v_{A,2}$ and rejects the call in cell $v_{A,3}$. Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{B,4}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm cannot accept any more calls. Assume, also, that the algorithm accepts the calls in the cells $v_{A,1}$ and $v_{A,3}$ (resp. $v_{A,2}$ and $v_{A,3}$) and rejects the call in cell $v_{A,2}$ (resp. $v_{A,1}$). Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{B,4}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm can only accept the call presented in cell $v_{C,2}$ (resp. $v_{C,1}$); this would give a benefit of at most 3.

Also, no algorithm could gain by accepting one of the four calls presented in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$. Assume that the algorithm rejects the calls in cells v_0, $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$, accepts the call in cell $v_{B,1}$ (resp. $v_{B,2}$, resp. $v_{B,3}$, resp. $v_{B,4}$), and rejects the calls in cells $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$ (resp. $v_{B,1}$, $v_{B,3}$, and $v_{B,4}$, resp. $v_{B,1}$, $v_{B,2}$, and $v_{B,4}$, resp. $v_{B,1}$, $v_{B,2}$, and $v_{B,3}$). Then, even if the adversary does not stop the sequence before producing the calls in the cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm can only accept the calls in the cells $v_{C,2}$, $v_{C,3}$ and $v_{C,4}$ (resp. $v_{C,1}$, $v_{C,3}$ and $v_{C,4}$, resp. $v_{C,1}$, $v_{C,4}$ and $v_{C,5}$, resp. $v_{C,2}$, $v_{C,3}$ and $v_{C,5}$); this would give a benefit of at most 4.

Also, no algorithm could gain by accepting two of the four calls presented in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$. Assume that the algorithm rejects the calls in the cells v_0, $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$, accepts the calls in the cells $v_{B,1}$ and $v_{B,2}$ (resp. $v_{B,2}$ and $v_{B,3}$, resp. $v_{B,1}$ and $v_{B,4}$) and rejects the calls in cells $v_{B,3}$ and $v_{B,4}$ (resp. $v_{B,1}$ and $v_{B,4}$, resp. $v_{B,2}$ and $v_{B,3}$). Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm could only accept the calls presented in cells $v_{C,3}$ and $v_{C,4}$ (resp. $v_{C,1}$ and $v_{C,4}$, resp. $v_{C,2}$ and $v_{C,3}$); this would give a benefit of at most 4. Similarly, assume that the algorithm rejects the calls in the cells v_0, $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$, accepts the calls in the cells $v_{B,1}$ and $v_{B,3}$ (resp. $v_{B,2}$ and $v_{B,4}$, resp. $v_{B,3}$ and $v_{B,4}$) and rejects the calls in cells $v_{B,2}$ and $v_{B,4}$ (resp. $v_{B,1}$ and $v_{B,3}$, resp. $v_{B,1}$ and $v_{B,2}$). Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm could only accept the call presented in cell $v_{C,4}$ (resp. $v_{C,3}$, resp. $v_{C,5}$); this would give a benefit of at most 3.

Also, no algorithm could gain by accepting three of the four calls presented in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$. Assume that the algorithm rejects the calls in the cells v_0, $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$, accepts the calls in the cells $v_{B,1}$, $v_{B,2}$ and $v_{B,3}$ (resp. $v_{B,1}$, $v_{B,2}$ and $v_{B,4}$) and rejects the calls in cells $v_{B,4}$ (resp. $v_{B,3}$). Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm could only accept the call presented in cell $v_{C,4}$ (resp. $v_{C,3}$); this would give a benefit of at most 4. Similarly, assume that the algorithm accepts the calls in the cells $v_{B,1}$, $v_{B,3}$ and $v_{B,4}$ (resp. $v_{B,2}$, $v_{B,3}$ and $v_{B,4}$) and rejects the calls in cells $v_{B,2}$ (resp. $v_{B,1}$). Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$,and $v_{C,5}$, the algorithm cannot accept any more calls. This would give a benefit of at most 3.

Also, no algorithm could gain by accepting all of the four calls presented in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$. Then, even if the adversary does not stop the sequence before producing the calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$, the algorithm cannot accept any more calls; this would give a benefit of at most 4.

The above observations can be easily made by studying carefully the subgraph of the interference graph induced by the thirteen cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{A,3}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, $v_{B,4}$, $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$ (see right part of Figure 3.14). Hence, we may assume that the algorithm either accepts all calls presented at a step or rejects them all.

Consider t executions of the algorithm on t sequences produced according to the probability distribution $\mathcal{P}$. Let q_0 be the number of executions in which A accepts the call produced in cell v_0, q_1 the number of executions in which A accepts the three calls in cells $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$, and q_2 the number of executions in which the algorithm accepts the four calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, and $v_{C,4}$.

The expected number of executions in which the algorithm does not accept the call in cell v_0 and the adversary produces calls in cells $v_{A,1}$, $v_{A,2}$ and $v_{A,3}$ is $\frac{1}{3}(t-q_0)$. Hence, the expected number of executions in which the algorithm does not accept the calls in cells v_0, $v_{A,1}$, $v_{A,2}$, and $v_{A,3}$ and the adversary produces calls in cells $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$ is $\frac{3}{4}\left(\frac{1}{3}(t-q_0)-q_1\right)$ and the expected number of executions in which the algorithm does not accept the calls in cells v_0, $v_{A,1}$, $v_{A,2}$, $v_{A,3}$, $v_{B,1}$, $v_{B,2}$, $v_{B,3}$, and $v_{B,4}$ and the adversary produces calls in cells $v_{C,1}$, $v_{C,2}$, $v_{C,3}$, $v_{C,4}$, and $v_{C,5}$ is $\frac{4}{5}\left(\frac{3}{4}\left(\frac{1}{3}(t-q_0)-q_1\right)-q_2\right)$. Thus,

$$\mathcal{E}_{\mathcal{P}}[B_A(\sigma)] \leq \frac{q_0+3q_1+4q_2+5\frac{4}{5}\left(\frac{3}{4}\left(\frac{1}{3}(t-q_0)-q_1\right)-q_2\right)}{t} = 1$$

and $c_A^{\mathcal{P}} \geq 127/60$. By Lemma 28, we obtain that this is a lower bound on the competitive ratio of any randomized algorithm against an oblivious adversary. □

The lower bound for planar networks follows. Note that the best known upper bound is 4 achieved by the Classify and Randomly Select algorithm [1, 52].

Theorem 32. *There exists a planar network on which no randomized call–control algorithm can be better than* 2.086*–competitive against oblivious adversaries.*

Proof. Consider a planar network with the interference graph shown in Figure 3.15. We will prove that there exists an adversary $\mathcal{ADV}$ that produces calls according to a probability distribution $\mathcal{P}$ in such way that no deterministic algorithm can be better than 2.086–competitive under $\mathcal{P}$ even if it knows the probability distribution $\mathcal{P}$ in advance.

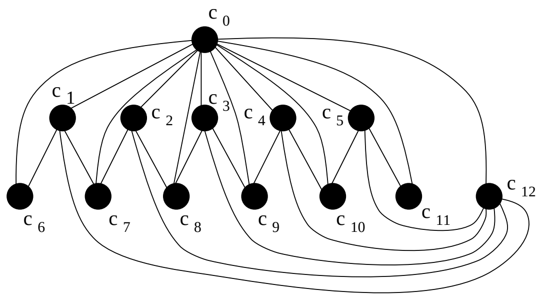

Figure 3.15: The planar interference graph used in the proof of Theorem 32.

We define the probability distribution $\mathcal{P}$ as follows. First, the adversary produces a call in cell c_0. Then, it

- either stops, with probability 4/5,
- or does the following, with probability 1/5. It presents a call in cells $c_1, ..., c_5$, and
  - either stops with probability 2/7,
  - or presents a call in cells $c_6, ..., c_{12}$, with probability 5/7.

It can be easily seen that the expected benefit of the optimal off–line algorithm on sequences of calls generated according to $\mathcal{P}$ is

$$\mathcal{E}_{\mathcal{P}}[B_{OPT}(\sigma)] = 1 \cdot \frac{4}{5} + 5 \cdot \frac{1}{5} \cdot \frac{2}{7} + 7 \cdot \frac{1}{5} \cdot \frac{5}{7} = \frac{73}{35}.$$

Let A be a deterministic call control algorithm that runs on the calls produced by $\mathcal{ADV}$. Consider t executions of the algorithm on t sequences produced according to the probability distribution $\mathcal{P}$. Let q_0 be the number of executions in which A accepts the call produced in cell c_0, and q_1 the number of executions in which A accepts both calls in cells c_1 and c_2.

The expected number of executions in which the algorithm does not accept the call in cell c_0 and the adversary produces calls in cells $c_1, ..., c_5$ is $\frac{1}{5}(t - q_0)$. Similarly, the expected number of executions in which the algorithm does not accept the calls in cells $c_0, c_1, ..., c_5$ and the adversary produces calls in cells $c_6, ..., c_{12}$ is $\frac{5}{7}\left(\frac{1}{5}(t - q_0) - q_1\right)$. Thus,

$$\mathcal{E}_{\mathcal{P}}[B_A(\sigma)] \leq \frac{q_0 + 5q_1 + 7\frac{5}{7}\left(\frac{1}{5}(t - q_0) - q_1\right)}{t} = 1,$$

and

$$c_A^{\mathcal{P}} \geq \frac{73}{35} = 2.086.$$

By Lemma 28, we obtain that this is a lower bound on the competitive ratio of any randomized algorithm against an oblivious adversary. $\square$

Chapter 4

Disk Graphs

4.1 Introduction

We study two fundamental graph problems, maximum independent set and minimum coloring. Given a graph G, the maximum independent set problem is to find an independent set (i.e., a set of nodes without edges between them) of maximum size, while the minimum coloring problem is to find an assignment of colors (i.e, positive integers) to the nodes of the graph so that no two nodes connected by an edge are assigned the same color and the number of colors used is minimized. We consider graphs modelling intersections of disks in the plane.

The intersection graph of a set of disks in the Euclidean plane is the graph having a node for each disk and an edge between two nodes if and only if the corresponding disks overlap. Each disk is defined by its radius and the coordinates of its center. Two disks overlap if the distance between their centers is strictly smaller than the sum of their radii. A graph G is called a *disk graph* if there exists a set of disks in the Euclidean plane whose intersection graph is G. The set of disks is called the disk representation of G. A disk graph is called unit disk graph if all disks in its disk representation have the same radius. A disk graph is σ-bounded if the ratio between the maximum and the minimum radius among all the disks in its disk representation is at most σ.

In disk graphs, maximum independent set and minimum coloring are important since they can model resource allocation problems in radio communication networks [29]. Consider a set of transmitters located in fixed positions within a geographical region. Each transmitter uses a specific frequency to transmit its messages. Two transmitters can successfully (i.e., without signal interference) transmit messages simultaneously either if they use different frequencies or if they use the same frequency and their ranges do not overlap. Given a set of transmitters in a radio network, in order to guarantee successful transmissions simultaneously, important engineering problems that have to be solved are the frequency assignment problem where the objective is to minimize the number of frequencies used all over the network, and the call admission problem where the objective is to find a maximum-sized set of transmitters which can use the same frequency. Assuming that all transmitters have circular range, the graph reflecting possible interference between pairs of transmitters is a disk graph. The frequency assignment and call admission problems are equivalent to minimum coloring and maximum independent set problems, respectively.

An instance of the maximum independent set or the minimum coloring problem may or may not include the disk representation (i.e., disk center coordinates and/or radii) of the disk graph as part of the input. Clearly, the latter case is more difficult. Information about the disk representation of a disk graph is not easy to extract. Actually, determining whether a graph is a disk graph is an NP-complete problem [31].

The maximum independent set in disk graphs has been proved to be NP-hard even for unit disk graphs and even if the disk representation is given as part of the input [10]. A naive independent set algorithm is the algorithm First-Fit: starting from an empty set, it incrementally constructs an independent set by examining the nodes of the graph in an arbitrary order and including a node in the independent set only if none of its neighbors has been previously included. When applied to unit disk graphs, First-Fit has approximation ratio at most 5 and does not use the disk representation [46] (also implicit in [32]). In [46], a 3-approximation algorithm is obtained by computing a specific ordering of the nodes of a unit disk graph and running First-Fit according to this ordering. A similar idea leads to a 5-approximation algorithm in general disk graphs [46]. Furthermore, as it has been observed in [20], a $(2.5 + \epsilon)$-approximation algorithm for unit disk graphs follows by a more general result presented in [30]. None of the algorithms above use the disk representation. Polynomial-time approximation schemes have been presented for both unit disk graphs [34, 45] and general disk graphs [22, 9] when the disk representation is given.

The minimum coloring problem has also been proved to be NP-hard in [10, 27] even for unit disk graphs. Again, First-Fit algorithm can be used. It examines the nodes of the graph in an arbitrary order and assigns to each node the smallest color not assigned to its already examined neighbors. Algorithm First-Fit computes 5-approximate solutions in unit disk graphs

[26, 46]. By processing the nodes of the graph in a specific order, First-Fit computes 3-approximate solutions in unit disk graphs [27, 46, 51]. In general disk graphs, a smallest-degree-last version of First-Fit achieves an approximation ratio of 5 [26, 44, 46].

In the on-line versions of the problems, the disk graph is not given in advance but is revealed in steps. In each step, a node of the graph appears together with its edges incident to nodes appeared in previous steps (and possibly, together with the center coordinates and/or the radius of the corresponding disk). When a node appears, an on-line independent set algorithm decides either to accept the node by including it in the independent set or to reject it, while an on-line coloring algorithm decides which color to assign to the node. In each case, the decisions of the algorithm cannot change in the future. The performance of an on-line algorithm is measured in terms of its competitive ratio (or competitiveness). For on-line independent set algorithms, the competitive ratio is defined as the maximum over all possible sequences of disks of the ratio of the size of the maximum independent set over the size of the independent set computed by the algorithm. For on-line coloring algorithms, the competitive ratio is defined as the maximum over all possible sequences of disks of the ratio of the number of colors used by the algorithm over the minimum number of colors sufficient for coloring the graph.

First-Fit is essentially an on-line algorithm. For the independent set problem, it has competitive ratio 5 in unit disk graphs [32, 46] and $O(\min\{n, \sigma^2\})$ in σ-bounded disk graphs with n nodes [20]. As it is observed in [20], First-Fit is optimal within the class of deterministic on-line algorithms.

The First-Fit coloring algorithm has been widely studied in a more general context and has been proved to be $\Theta(\log n)$-competitive in inductive graphs with n nodes [35, 28]. The lower bound holds also for trees (which are disk graphs) so the $\Theta(\log n)$ bound holds for general disk graphs. In unit disk graphs, First-Fit is at most 5-competitive [26, 46] while for σ-bounded disk graphs with n nodes, it is at most $O(\min\{\log n, \sigma^2\})$-competitive [19]. For unit disk graphs, a lower bound of 2 on the competitiveness of any deterministic on-line coloring algorithm is presented in [24]. The best known lower bound on the competitiveness of deterministic coloring algorithms in σ-bounded disk graphs is $\Omega(\min\{\log n, \log\log\sigma\})$ [19]. Better on-line coloring algorithms exist for σ-bounded disk graphs in the case where the disk representation is given. Most of them use First-Fit as a subroutine. The best competitiveness upper bound in this case is $O(\min\{\log n, \log\sigma\})$ [19].

In this chapter, we study the on-line version of both problems. For the independent set problem, we investigate whether randomization helps in improving the competitiveness of on-line algorithms, also addressing the cellular network model, presented in previous chapters. For randomized on-line independent set algorithms, the competitive ratio is defined as the maximum over all possible sequences of disks of the ratio of the size of the maximum independent set over the expected size of the independent set computed by the algorithm. We assume that the sequences of disks are selected by oblivious adversaries, i.e., adversaries that have no knowledge of the random choices of the algorithms (but may know the probability distribution used by the algorithm for making random choices). This is a typical assumption usually made in the study of randomized on-line algorithms [7]. Somewhat surprisingly, we show that, in general, randomization does not help against oblivious adversaries even if the disk representation is given, i.e., we construct sequences of disks for which no (possibly randomized) on-line algorithm can be better than $\Omega(n)$-competitive. In the case that the disk representation is not given, we prove a lower bound of $\Omega(\min\{n, \sigma^2\})$ on the competitiveness of on-line algorithms on σ-bounded disk graphs with n nodes meaning that algorithm First-Fit is optimal within a small constant factor. For the case of σ-bounded disk graphs with given representation, we present randomized algorithms with competitive ratio almost logarithmic in σ and show that they are optimal. For unit disk graphs, we present a randomized algorithm with competitive ratio 4.41. We also show lower bounds of 2.5 and 3 for randomized algorithms in unit disk graphs. Our results for the on-line independent set problem together with the previously known results on deterministic on-line algorithms are summarized in Table 4.1. For the coloring problem, we show how to achieve the best known upper bound of $O(\min\{\log n, \log \sigma\})$ for σ-bounded sequences of n disks even if the disk representation is not given.

		Deterministic algorithms	Randomized algorithms		
	Disk repr.	Lower/upper bound	Lower bound	Upper bound	Alg.
σ-DG	Yes	$\Theta(\min\{n, \sigma^2\})$	* $\Omega(\min\{n, \log \sigma\})$	* $O(\min\{n, \log \sigma\})$ * $O\left(\min\{n, \frac{1}{\epsilon} \log \sigma \log^{1+\epsilon} \log \sigma\}\right)$	Classify Guess
σ-DG	No	$\Theta(\min\{n, \sigma^2\})$	* $\Omega(\min\{n, \sigma^2\})$	$O(\min\{n, \sigma^2\})$	First-Fit
UDG	Yes	5	* 2.5	* $\frac{8\sqrt{3}}{\pi} \approx 4.41$	Filter
UDG	No	5	* 3	5	First-Fit

Table 4.1: Summary of results for the on-line independent set problem. (*) indicates results in this PhD Thesis.

The rest of the chapter is structured as follows. Section 4.2 is devoted to the on-line independent set problem in σ-bounded disk graphs. Our results for unit disk graphs are presented in Section 4.3 while our coloring algorithm is presented in Section 4.4.

4.2 Independent sets in σ-bounded disk graphs

4.2.1 Upper Bounds

In this section we present the randomized on-line algorithm Classify for computing independent sets in disk graphs. It has a competitive ratio $O(\min\{n, \log \sigma\})$ against oblivious adversaries on σ-bounded disk graphs with n nodes. The algorithm uses the value of σ which is supposed to be known in advance and makes its random choices based on the disk representation. Despite these limitations, this is the first algorithm achieving a competitive ratio logarithmic in σ and (as we will prove in Section 4.2.2) is optimal among the on-line algorithms that use the disk representation.

Algorithm Classify works as follows. When the first disk is presented, the algorithm flips a coin. On heads, it accepts the disk and executes algorithm First-Fit for disks having radii in the interval $[R, 2R)$, where R is the radius of the first disk presented, ignoring (i.e., rejecting) all other disks. On tails, the algorithm selects equiprobably a number i from the set $\{-\lceil \log \sigma \rceil, -\lceil \log \sigma \rceil + 1, ..., -1, 1, ..., \lceil \log \sigma \rceil\}$ and executes algorithm First-Fit for disks of radius in the interval $[R2^i, R2^{i+1})$, ignoring (i.e., rejecting) all other disks.

We prove the following theorem.

Theorem 33. *Algorithm* Classify *is* $O(\min\{n, \log \sigma\})$*-competitive against oblivious adversaries on* σ*-bounded disk graphs with n nodes.*

Proof. Since the first disk is accepted with probability 1/2, the algorithm has competitive ratio $O(n)$. In what follows, we show that the algorithm is $O(\log \sigma)$-competitive as well. Denote by OPT the optimal independent set of the sequence. For $i = -\lceil \log \sigma \rceil, -\lceil \log \sigma \rceil - 1, ..., \lceil \log \sigma \rceil$, denote by S_i the set of disks with radius in the interval $[R2^i, R2^{i+1})$ and by OPT_i the maximum independent set among the disks belonging to set S_i. Clearly, $|OPT_i| \geq |OPT \cap S_i|$ since $OPT \cap S_i$ is an independent set for S_i. Assume that the algorithm selects set S_i and executes algorithm First-Fit on the disks of that set. Observe that disks in S_i form a 2-bounded disk graph. In such graphs, the following lemma gives a guarantee on the performance of algorithm First-Fit for computing independent sets.

Lemma 34. *Algorithm* First-Fit *is at most* 15*-competitive on* 2*-bounded disk graphs.*

Proof. Actually, we provide an alternative proof that First-Fit is at most $O(\sigma^2)$-competitive on σ-bounded disk graphs which allows to show that the hidden constant in the $O(\sigma^2)$ notation is small.

Consider the application of algorithm First-Fit on a graph. The number of optimal nodes that may be blocked by a node accepted by First-Fit is at most the size of the maximum independent set in its neighborhood. We will show that no node of a σ-bounded disk graph has an independent set of size at least $(\sigma+2)^2$ in its neighborhood. Therefore, the competitive ratio of First-Fit on σ-bounded disk graphs is at most the largest integer strictly smaller than $(\sigma+2)^2$. In the case of $\sigma = 2$, we obtain that First-Fit is at most 15-competitive.

Consider a disk D_0 centered at point C_0 with radius R and assume that there exists a set S of at least $(\sigma+2)^2$ mutually non-overlapping disks which overlap with D_0. Let $D_1, D_2, ..., D_{|S|}$ be the disks in S. For the disk D_i ($1 \leq i \leq |S|$), denote by C_i its center, by r_i its radius and by d_i the distance of its center C_i from point C_0.

Set $r_{\min} = \min_{0\leq i\leq |S|}\{r_i\}$. Observe that S contains at most one disk D_j which may contain point C_0 (i.e., $d_j < r_j$), otherwise the disks in set S would not be non-overlapping. If such a disk D_j exists then, certainly, the intersection of D_j and D_0 contains a disk of radius $r_{\min}$.

Now define $R' = 2r_{\min} + \max_{1\leq i\leq |S|}\{d_i - r_i\}$ and let D' be the disk centered at C_0 with radius R'. For each disk D_i not containing C_0 (i.e., $d_i \geq r_i$), consider the disk of radius $r_{\min}$ centered at the point in the segment C_0C_i which is at distance $d_i - r_i + r_{\min}$ from C_0. This disk is completely contained in the intersection of D_i and D'.

So, the total area of the intersections of disks in S with D' is at least

$$\begin{aligned}
(\sigma+2)^2\pi r_{\min}^2 &\geq \left(\frac{R}{r_{\min}}+2\right)^2 \pi r_{\min}^2 \\
&= \pi(2r_{\min}+R)^2 \\
&> \pi\left(2r_{\min} + \max_{1\leq i\leq |S|}\{d_i - r_i\}\right)^2 \\
&= \pi R'^2.
\end{aligned}$$

The strict inequality follows by the fact that all disks in S overlap with D_0. We have obtained that the total area of the intersections of disks in S with disk D' is larger than the area of D', which contradicts the assumption that the disks in S are mutually non-overlapping. □

Let B be the number of disks accepted by algorithm Classify and let B_i

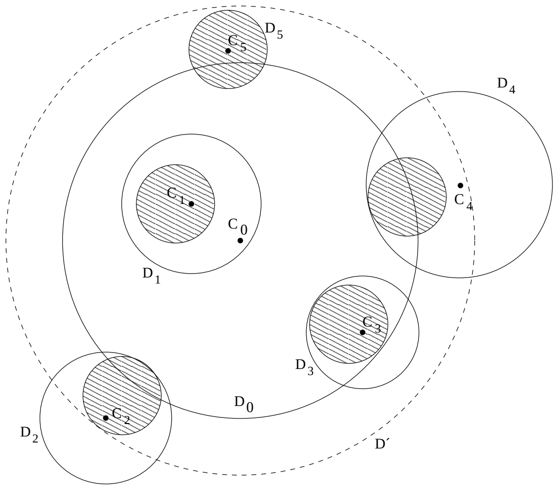

Figure 4.1: The disk D_0 and 5 mutually non-overlapping disks which overlap with D_0. Grey disks indicate that the disk with the smallest radius is completely contained within disks $D_1, ..., D_5$.

be the number of disks accepted by the algorithm if we assume that it selects the set S_i and executes algorithmFirst-Fit only for disks of the selected set.

Using the lemma, we obtain that the algorithm accepts at least

$$B_i \geq \frac{1}{15}|OPT_i| \geq \frac{1}{15}|OPT \cap S_i|$$

disks of S_i. Now, the expected size of the independent set computed by algorithm Classify is

$$\begin{aligned} E[B] &= \sum_{i=-\lceil \log \sigma \rceil}^{\lceil \log \sigma \rceil} (\Pr[S_i \text{ is selected}] \cdot B_i) \\ &\geq \frac{1}{15} \sum_{i=-\lceil \log \sigma \rceil}^{\lceil \log \sigma \rceil} (\Pr[S_i \text{ is selected}] \cdot |OPT \cap S_i|) \\ &\geq \frac{1}{15} \min_i \{\Pr[S_i \text{ is selected}]\} \cdot |OPT| \\ &\geq \frac{1}{60\lceil \log \sigma \rceil} \cdot |OPT|. \end{aligned}$$

Hence, the competitive ratio of the algorithm is $O(\log \sigma)$. □ □

We now present algorithm Guess which achieves a slightly weaker competitive ratio but does not need to known neither n nor σ in advance. Consider a sequence of n disks and let R be the radius of the first disk of the sequence. Then, for any $i = 0, 1, ..., 2\lceil \log \sigma \rceil - 1$, define the set of disks S_i with radii at least $R/2^{\lceil \log \sigma \rceil - i}$ and smaller than $R/2^{\lceil \log \sigma \rceil - i - 1}$. When the first disk of each set is presented, the algorithm probabilistically determines whether it will consider disks from that specific set and ignore all disks from all other sets.

We define the function $\alpha : [1, +\infty) \to R^+$ as follows:

$$\alpha(x) = \frac{1}{\log e}(\lfloor \log x \rfloor + 1)\alpha(\lfloor \log x \rfloor), \qquad \text{if } x \geq 2$$

$$\alpha(x) = \frac{2 \log e}{4 \log e - 5}, \quad \text{if } 2 > x \geq 1$$

When the first disk of the i-th set appears, algorithm Guess tosses a fair coin with

$$\Pr[\text{HEADS}] = \begin{cases} 1/2 & \text{for } i \in \{1, 2\} \\ \frac{1}{\alpha(i-2)(i-1)} & \text{for } i \geq 3 \end{cases}$$

If the outcome is HEADS, the algorithm decides to execute algorithm First-Fit for all disks of this set and ignores (i.e., rejects) the disks of all the other sets; if the outcome is TAILS, the algorithm rejects all disks of this set.

Theorem 35. *Algorithm* Guess *is at most* $O\left(\min\left\{n, \prod_{j=1}^{\log^* \sigma - 1} \log^{(j)} \sigma\right\}\right)$, *against oblivious adversaries in σ-bounded disk graphs with n nodes.*

Proof. First, observe that algorithm Guess accepts the first disk of the sequence with probability 1/2. So, the competitive ratio of the algorithm is $O(n)$. In the following, we will show that it is also $O\left(\prod_{j=1}^{\log^* \sigma - 1} \log^{(j)} \sigma\right)$.

The proof proceeds in a similar way to the one of Theorem 33. Denote by OPT the maximum independent set of the sequence and let $\kappa \leq 1 + \lceil \log \sigma \rceil$ be the number of all different sets. For all $i = 0, 1, ..., \kappa$, denote by S'_i the i-th set, by OPT_i the maximum independent set for the disks of set S'_i, and by $\mathcal{E}_i$ the fact stating that algorithm Guess decides to execute algorithm First-Fit for the disks of the i-th set. It is: $|OPT_i| \geq |OPT \cap S'_i|$, since the set $OPT \cap S'_i$ is an independent set for S'_i. We assume that the algorithm selects the set S'_i and executes algorithm First-Fit for the disks of this set. Since, for all $i = 1, ..., \kappa$, disks of set S'_i form a 2-bounded disk graph, assuming that the algorithm selects the set S'_i and executes algorithm First-Fit for the disks of this set, by Lemma 34, we conclude that the algorithm accepts

$$B_i \geq \frac{1}{15} |OPT_i| \geq \frac{1}{15} |OPT \cap S'_i|$$

disks of set S_i'. The expected size of the independent set computed by algorithm Guess is:

$$\begin{aligned} E[B] &= \sum_{i=1}^{\kappa} (\Pr[\mathcal{E}_i] \cdot B_i) \\ &\geq \frac{1}{15} \sum_{i=1}^{\kappa} (\Pr[\mathcal{E}_i] \cdot |OPT \cap S_i) \\ &\geq \frac{1}{15} \min_i \{\Pr[\mathcal{E}_i]\} \cdot |OPT|. \end{aligned}$$

So, for showing that the competitive ratio of algorithm Guess is $O\left(\prod_{j=1}^{\log^* \sigma - 1} \log^{(j)} \sigma\right)$, we will show that for all $i = 1, ..., \kappa$, it is $\Pr[\mathcal{E}_i] \in \Omega\left(\frac{1}{\prod_{j=1}^{\log^* \sigma - 1} \log^{(j)} \sigma}\right)$.

For all $i = 1, ..., \kappa$, the probability that the event $\mathcal{E}_i$ occurs equals the probability that all disks belonging to the first $i - 1$ sets (if there exist) are rejected times the probability that the i-th set is selected. Obviously, $\Pr[\mathcal{E}_1] = 1/2$, $\Pr[\mathcal{E}_2] = 1/4$, $\Pr[\mathcal{E}_3] = \frac{4\log e - 5}{16 \log e}$, while for $i \geq 4$, it is

$$\Pr[\mathcal{E}_i] = \frac{1}{4}\left(\prod_{j=1}^{i-3}\left(1 - \frac{1}{\alpha(j)(j+1)}\right)\right)\frac{1}{\alpha(i-2)(i-1)}.$$

For bounding from below this probability, we use the following three technical claims.

Claim 36. *For all $\beta \in [1, +\infty)$ and integer $i > 1$, it is:*

$$\prod_{j=i_1}^{i_2}\left(1 - \frac{1}{\beta(j+1)}\right) \geq \left(\frac{i_1}{i_2+1}\right)^{\frac{1}{\beta}}.$$

Proof. First, we show that for all $j \geq 1$, it is

$$\beta\left(1 + \frac{1}{j}\right) - \frac{1}{j} \quad \geq \quad \beta\left(1 + \frac{1}{j}\right)^{1 - \frac{1}{\beta}} \tag{4.1}$$

Consider the function $F : [1, +\infty] \rightarrow R$ defined as

$$F(\beta) = \beta\left(1 + \frac{1}{j}\right) - \frac{1}{j} - \beta\left(1 + \frac{1}{j}\right)^{1 - \frac{1}{\beta}}.$$

Its first derivative is

$$
\begin{aligned}
F'(\beta) &= \left(1+\frac{1}{j}\right) - \left(1+\frac{1}{j}\right)^{1-\frac{1}{\beta}} - \frac{1}{\beta}\ln\left(1+\frac{1}{j}\right)\left(1+\frac{1}{j}\right)^{1-\frac{1}{\beta}} \\
&= \left(1+\frac{1}{j}\right)^{1-\frac{1}{\beta}}\left(\left(1+\frac{1}{j}\right)^{\frac{1}{\beta}} - 1 - \frac{1}{\beta}\ln\left(1+\frac{1}{j}\right)\right) \\
&\geq 0.
\end{aligned}
$$

The last inequality stems from the inequality $x \geq 1 + \ln x$, for all $x \geq 1$. Since the function $F(\cdot)$ is non-increasing and $F(1) = 0$, it is implied that $F(\beta) \geq 0$, for all $\beta \in [1, +\infty)$, and, thus, (4.1) holds.

Using (4.1) we have:

$$
\begin{aligned}
\prod_{j=i_1}^{i_2}\left(1 - \frac{1}{\beta(j+1)}\right) &= \frac{1}{\beta^{i_2-i_1+1}}\prod_{j=i_1}^{i_2}\frac{\beta j + \beta - 1}{j+1} \\
&= \frac{1}{\beta^{i_2-i_1+1}}\left(\prod_{j=i_1}^{i_2}\left(\beta\left(1+\frac{1}{j}\right) - \frac{1}{j}\right)\right)\prod_{j=i_1}^{i_2}\frac{j}{j+1} \\
&\geq \frac{i_1}{(i_2+1)\beta^{i_2-i_1+1}}\prod_{j=i_1}^{i_2}\left(\beta\left(1+\frac{1}{j}\right)^{1-\frac{1}{\beta}}\right) \\
&= \frac{i_1}{i_2+1}\left(\prod_{j=i_1}^{i_2}\frac{j+1}{j}\right)^{1-\frac{1}{\beta}} \\
&= \left(\frac{i_1}{i_2+1}\right)^{\frac{1}{\beta}}
\end{aligned}
$$

□

Claim 37. *For all $i > 0$, it is*

$$\prod_{j=1}^{i}\left(1 - \frac{1}{\alpha(j)(j+1)}\right) \geq \left(1 - \frac{1}{2\alpha(1)}\right)^{\log^* i + 1}.$$

Proof. First, we will show that

$$\prod_{j=1}^{i}\left(1 - \frac{1}{\alpha(j)(j+1)}\right) \geq \left(1 - \frac{1}{2\alpha(1)}\right)\prod_{j=1}^{\lfloor \log i \rfloor}\left(1 - \frac{1}{\alpha(j)(j+1)}\right). \quad (4.2)$$

The lemma will result from applying inequality (4.2) recursively.

Claim 36 and the definition of the function $\alpha(\cdot)$, imply that:

$$\begin{aligned}
\prod_{j=1}^{i}\left(1-\frac{1}{\alpha(j)(j+1)}\right) &\geq \prod_{j=0}^{\lfloor \log i\rfloor}\prod_{j'=2^j}^{2^{j+1}-1}\left(1-\frac{1}{\alpha(j')(j'+1)}\right)\\
&= \left(1-\frac{1}{2\alpha(1)}\right)\prod_{j=1}^{\lfloor \log i\rfloor}\prod_{j'=2^j}^{2^{j+1}-1}\left(1-\frac{1}{\alpha(j')(j'+1)}\right)\\
&\geq \left(1-\frac{1}{2\alpha(1)}\right)\prod_{j=1}^{\lfloor \log i\rfloor}\left(\frac{1}{2}\right)^{\frac{1}{\alpha(2^j)}}\\
&= \left(1-\frac{1}{2\alpha(1)}\right)\prod_{j=1}^{\lfloor \log i\rfloor}\left(\frac{1}{2}\right)^{\frac{\log e}{\alpha(j)(j+1)}}\\
&= \left(1-\frac{1}{2\alpha(1)}\right)\prod_{j=1}^{\lfloor \log i\rfloor} e^{-\frac{1}{\alpha(j)(j+1)}}\\
&\geq \left(1-\frac{1}{2\alpha(1)}\right)\prod_{j=1}^{\lfloor \log i\rfloor}\left(1-\frac{1}{\alpha(j)(j+1)}\right),
\end{aligned}$$

where the last inequality stems from the fact that $e^{-x} \geq 1-x$, for all $x \geq 0$. ☐

Claim 38. *For all $i \geq 1$, it is*

$$\alpha(i) \leq \left(\frac{5}{4\log e}\right)^{\log^* i-2}\frac{4}{\log^2 e}\alpha(1)\prod_{j=1}^{\log^* i}\log^{(j)} i.$$

Proof. Obviously, if $\log^* i = 0$, i,e., $i \in [1,2)$, then it is

$$\alpha(i) = \alpha(1) \leq \frac{64}{25}\alpha(1) = \left(\frac{5}{4\log e}\right)^{-2}\frac{4}{\log^2 e}\alpha(1).$$

If $\log^* i = 1$, i.e., $i \in [2, 4)$, then it is

$$\begin{aligned}
\alpha(i) &= \frac{1}{\log e}\alpha(\lfloor \log i \rfloor)(\lfloor \log i \rfloor + 1) \\
&\leq \frac{2}{\log e}\alpha(1)\log i \\
&\leq \frac{16}{5\log e}\alpha(1)\log i \\
&= \left(\frac{5}{4\log e}\right)^{-1} \frac{4}{\log^2 e}\alpha(1)\log i.
\end{aligned}$$

Similarly, if $\log^* i = 2$, i.e., $i \in [4, 16)$, then

$$\begin{aligned}
\alpha(i) &= \frac{1}{\log e}\alpha(\lfloor \log i \rfloor)(\lfloor \log i \rfloor + 1) \\
&\leq \frac{1}{\log^2 e}\alpha(\lfloor \log \lfloor \log i \rfloor \rfloor)(\lfloor \log i \rfloor + 1)(\lfloor \log \lfloor \log i \rfloor \rfloor + 1) \\
&\leq \frac{4}{\log^2 e}\alpha(1)\log i \log\log i \\
&= \left(\frac{5}{4\log e}\right)^{-1} \frac{4}{\log^2 e}\alpha(1)\log i \log\log i.
\end{aligned}$$

We assume that $\log^* i > 2$, i.e., $i \geq 16$. Then, it is:

$$\begin{aligned}
\alpha(i) &= \frac{1}{\log e}\alpha(\lfloor \log i \rfloor)(\lfloor \log i \rfloor + 1) \\
&\leq \frac{5}{4\log e}\alpha(\log i)\log i \\
&\leq \ldots \\
&\leq \left(\frac{5}{4\log e}\right)^{\log^* i - 2} \alpha(\log^{(\log^* i - 2)} i) \prod_{j=1}^{\log^* i - 2} \log^{(j)} i \\
&\leq \left(\frac{5}{4\log e}\right)^{\log^* i - 2} \frac{4}{\log^2 e}\alpha(1) \prod_{j=1}^{\log^* i} \log^{(j)} i.
\end{aligned}$$

The inequality in the second row stems form the fact that the function $\alpha(\cdot)$ is non-increasing and from the observation that for all $i' \geq 16$, it is $\lfloor \log i' \rfloor + 1 \leq \frac{5}{4}\log i'$. The third row results from recursive application of this inequality, while the last inequality hold since $\log^{(\log^* i - 2)} i \in [4, 16)$. Thus, the proof of the Claim is completed. $\square$

Using Claims 37 and 38 and replacing $\alpha(1) = \frac{2\log e}{4\log e - 5}$, we have

$$\Pr[\mathcal{E}_i] =$$
$$\frac{1}{4}\left(\prod_{j=1}^{i-3}\left(1 - \frac{1}{\alpha(j)(j+1)}\right)\right)\frac{1}{\alpha(i-2)(i-1)} \geq$$
$$\frac{1}{4}\left(1 - \frac{1}{2\alpha(1)}\right)^{\log^*(i-3)+1}\frac{1}{\alpha(i-2)(i-1)} \geq$$
$$\frac{1}{4}\left(\frac{5}{4\log e}\right)^{\log^*(i-3)+1}\frac{1}{(i-1)\left(\frac{5}{4\log e}\right)^{\log^*(i-2)-2}\frac{4}{\log^2 e}\alpha(1)\prod_{j=1}^{\log^*(i-2)}\log^{(j)}(i-2)} \geq$$
$$\frac{125(4\log e - 5)}{2048\log^2 e(i-1)\prod_{j=1}^{\log^*(i-2)}\log^{(j)}(i-2)}$$

Since the number of all different sets κ is at most $1 + \lceil\log\sigma\rceil < 2 + \log\sigma$, we have

$$\Pr[\mathcal{E}_i] \in \Omega\left(\frac{1}{\prod_{j=1}^{\log^*\sigma-1}\log^{(j)}\sigma}\right)$$

thus, completing the proof of the theorem. $\square$

4.2.2 Lower bounds

The lower bounds presented in this section show that, in general, randomization does not help, i.e., there are sequences of n disks for which any on-line algorithm is $\Omega(n)$-competitive even if the disk representation is given. For σ-bounded disk graphs, the next lower bound states that when the disk representation is not given, on-line algorithms with competitive ratio logarithmic in σ do not exist.

Theorem 39. *Any randomized on-line algorithm for computing independent sets in σ-bounded disk graphs with n nodes is $\Omega(\min\{n, \sigma^2\})$-competitive against oblivious adversaries, if the disk representation is not given.*

Proof. Let κ be a positive integer. We will construct an adversary which generates a graph G_κ with an independent set of size $\kappa + 1$ such that the expectation of the size of the independent set of G_κ that any randomized on-line algorithm can find is at most 2.

The graph G_κ generated by the adversary is defined as follows. The nodes of G_κ are partitioned into κ levels $0, 1, ..., \kappa - 1$. Each level i has two nodes: a left node v_l^i and a right node v_r^i. The two nodes of a level are non-adjacent. First, the adversary generates the two nodes of level 0. For $i = 1, ..., \kappa - 1$, the nodes of level i are generated after the nodes of level $i - 1$. The adversary tosses a coin in order to connect the nodes of level i with nodes of smaller levels. On heads, it connects both nodes of level i to node v_l^{i-1} and to all nodes of levels $i-2, i-3, ..., 0$ to which node v_l^{i-1} is connected; on tails, it connects both nodes of level i to node v_r^{i-1} and to all nodes of levels $i-2, ..., 0$ to which node v_r^{i-1} is connected.

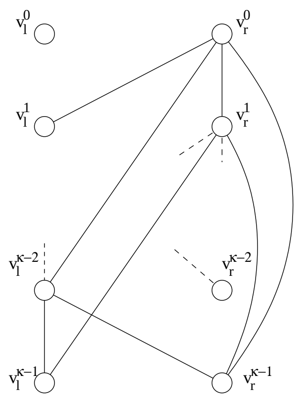

Figure 4.2: An exmple for graph G_κ.

Consider the set of nodes consisting of the two nodes of level $\kappa - 1$ and, for $i = 0, ..., \kappa - 2$, of the node of level i which is not connected to nodes of higher levels. This is an independent set of G_κ. Hence, the optimal independent set of G_κ has size at least $\kappa + 1$.

In what follows we will show that the size of the independent set of G_κ any (possibly randomized) on-line algorithm can compute is at most 2. Consider the application of an algorithm A on t sequences of disks produced by the adversary. Denote by l_i the number of executions in which the algorithm accepts the left node of level i, by r_i the number of executions in which the algorithm accepts the right node of level i, and by b_i the number of executions in which the algorithm accepts both nodes of level i.

For $i = 0, 1, ..., \kappa - 1$, let X_i be the random variable denoting the number of executions in which the nodes presented at level i are unconstrained by nodes of smaller levels (i.e., they are not connected to nodes of smaller levels that have been accepted by the algorithm). Then, nodes of level $i + 1$ are constrained only if:

- the left node of level i is rejected and the nodes of level $i + 1$ are connected to the left node of level i or

- the right node of level i is rejected and the nodes of level $i+1$ are connected to the right node of level i.

Hence,

$$E[X_{i+1}|X_i] \leq X_i - \frac{r_i + l_i}{2} - b_i$$

and

$$\begin{aligned} E[X_{\kappa-1}] &\leq t - \sum_{i=0}^{\kappa-2} \frac{r_i + l_i + 2b_i}{2} \Rightarrow \\ \sum_{i=0}^{\kappa-2} (r_i + l_i + 2b_i) &\leq 2t - 2E[X_{\kappa-1}] \end{aligned} \tag{4.3}$$

Since the number of executions in which the algorithm accepts at least one node from level $\kappa-1$ is at most the number of executions in which the nodes of level $\kappa-1$ are unconstrained, it is $r_{\kappa-1} + l_{\kappa-1} + b_{\kappa-1} \leq E[X_{\kappa-1}]$. Now, using (4.3), we obtain that the expectation of the size of the independent set $B(G_\kappa)$ computed by the algorithm is

$$\begin{aligned} E[|B(G_\kappa)|] &= \frac{1}{t}\sum_{i=0}^{\kappa-1}(r_i + l_i + 2b_i) \\ &= \frac{1}{t}\sum_{i=0}^{\kappa-2}(r_i + l_i + 2b_i) + \frac{1}{t}(r_{\kappa-1} + l_{\kappa-1} + 2b_{\kappa-1}) \\ &\leq 2 - \frac{1}{t}(2E[X_{\kappa-1}] - (r_{\kappa-1} + l_{\kappa-1} + 2b_{\kappa-1})) \\ &\leq 2 - \frac{2}{t}(E[X_{\kappa-1}] - (r_{\kappa-1} + l_{\kappa-1} + b_{\kappa-1})) \\ &\leq 2. \end{aligned}$$

We conclude that the competitive ratio of the algorithm is at least $\frac{\kappa+1}{2}$.

It remains to show that graph G_κ for $\kappa = \Omega\left(\min\{n, \sigma^2\}\right)$ is a σ-bounded disk graph.

Lemma 40. *For any $\sigma \geq 2$ and $n \geq 8$, graph G_{4d^2} for*

$$d = \min\left\{\left\lfloor\frac{\sigma+2}{4}\right\rfloor, \left\lfloor\sqrt{\frac{n}{8}}\right\rfloor\right\}$$

is a σ-bounded disk graph with at most n nodes.

Proof. Consider $\sigma \geq 2$ and an integer $n \geq 8$. We define as

$$d = \min\left\{\left\lfloor\frac{\sigma+2}{4}\right\rfloor, \left\lfloor\sqrt{\frac{n}{8}}\right\rfloor\right\}.$$

Obviously, graph G_{4d^2} has $8d^2 \leq n$ vertices. We will show that the graph G_{4d^2} is a σ-bounded disk graph.

We construct the following disk representation of G_{4d^2}. The set consists of $8d^2$ disks and is partitioned into d disjoint sets called rings. For $i = 1, ..., d$, ring i contains $8i-4$ disks called planets and $8i-4$ disks called satellites. All planets are centered at the same point and all satellites have unit radius.

In each ring i, the j-th planet (for $j = 0, ..., 8i-5$) has radius $4(i-1) + 1 + \frac{j}{8i-5}$. Observe that all planets have radii between 1 and $4d-2 \leq \sigma$. Hence, the disk graph we construct is σ-bounded. We will also show that it is a disk representation of G_{4d^2}.

For locating the satellites of ring i, pick $8i-4$ lines originating from the center of the planets which partition the plane into $8i-4$ equal sectors. The center of the j-th satellite of ring i is located on the bisection of sector j and distance $4(i-1) + 2 + \frac{2j+1}{16i-10}$ from the center of the planets. The construction for $d = 3$ is depicted in Figure 4.3.

Consider graph G_{4d^2} and let v_p^i be the node of level i which is connected to nodes of higher levels. Denote by v_s^i the other node of level i. We will show that the set of disks we constructed is a disk representation of G_{4d^2}. In particular, we will map each node of graph G_{4d^2} to a disk and we will show that, for each pair of nodes connected with an edge in G_{4d^2}, the corresponding disks overlap and that, for each pair of nodes not connected with an edge in G_{4d^2}, the corresponding disks do not overlap.

Consider the function $\ell(i,j)$ which denotes the position of the j-th planet of ring i in the descending ordering of the planets according to their radius. Since (i) each ring contains equal same number of planets and satellites, (ii) the j-th planet of ring i has larger radius than planets $0, ..., j-1$ of ring i and all planets of rings $1, .., i-1$, and (iii) the j-th satellite of ring i has larger distance from the center of the planets than satellites $0, ..., j-1$ of ring i and all satellites of rings $1, .., i-1$, $\ell(i,j)$ also denotes the position of the j-th satellite of ring i in the descending ordering of the satellites according to their distance from the center of the planets. Clearly, if $\ell(i_1, j_1) \geq \ell(i_2, j_2)$, then it is either $i_1 \leq i_2$ or $i_1 = i_2$ and $j_1 \leq j_2$.

We map the j-th planet of ring i to node $v_p^{\ell(i,j)}$ of G_{4d^2} and the j-th satellite of ring i to node $v_s^{\ell(i,j)}$ of G_{4d^2}.

Consider a pair of nodes $v_p^{\xi_1}$ and $v_p^{\xi_2}$ connected with an edge in G_{4d^2}.

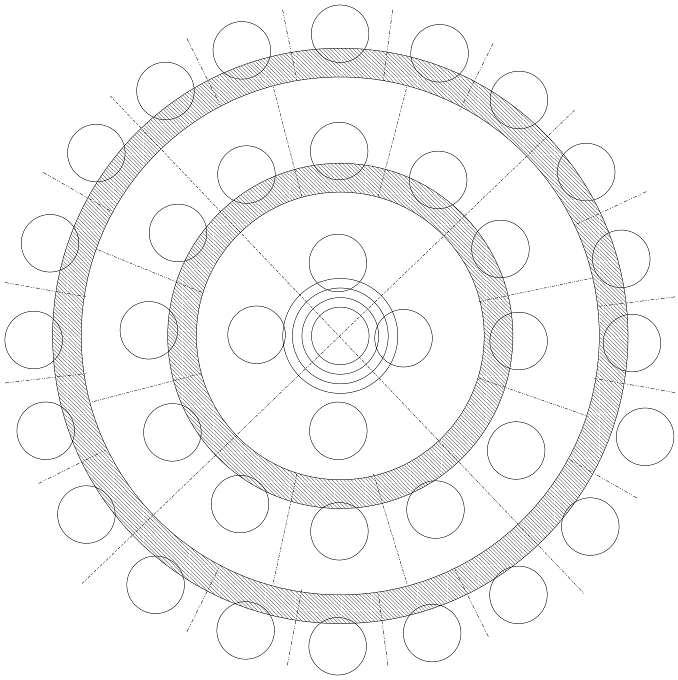

Figure 4.3: The construction of the sequences of disks generated by the adversary for the proof of Lemma 6. Grey rings denote the peripheries of planets of the same rings.

The corresponding disks are planets and certainly overlap since they have common center.

Consider a pair of nodes $v_p^{\xi_1}$ and $v_s^{\xi_2}$ connected with an edge in G_{4d^2}. Then, by the definition of graph G_{4d^2}, it must be $\xi_1 < \xi_2$. The corresponding disks are the j_1-th planet of ring i_1 and the j_2-th satellite of ring i_2 such that $\ell(i_1, j_1) = \xi_1$ and $\ell(i_2, j_2) = \xi_2$. Since $\xi_1 < \xi_2$, it is either $i_1 > i_2$ or $i_1 = i_2$ and $j_1 > j_2$. In the first case, the sum of the radii of the j_1-th planet of ring i_1 and the j_2-th satellite of ring i_2 is

$$\begin{aligned} 4(i_1-1)+2+\frac{j_1}{8i_1-5} &\geq 4i_2+2 \\ &> 4(i_2-1)+2+\frac{2j_2+1}{16i_2-10} \end{aligned}$$

which is the distance of the center of the j_2-th satellite of ring i_2 from the center of the planets. In the second case, the sum of the radii of the j_1-th planet and the j_2-th satellite of ring i_1 is

$$\begin{aligned} 4(i_1-1)+2+\frac{j_1}{8i_1-5} &\geq 4(i_1-1)+2+\frac{j_2+1}{8i_1-5} \\ &> 4(i_2-1)+2+\frac{2j_2+1}{16i_2-10} \end{aligned}$$

which is the distance of the center of the j_2-th satellite of ring i_2 from the center of the planets. Hence, in both cases, the two disks overlap.

Consider a pair of nodes $v_p^{\xi_1}$ and $v_s^{\xi_2}$ not connected with an edge in G_{4d^2}. By the definition of G_{4d^2}, it must be $\xi_1 \geq \xi_2$. The corresponding disks are the j_1-th planet of ring i_1 and the j_2-th satellite of ring i_2 such that $\ell(i_1, j_1) = \xi_1$ and $\ell(i_2, j_2) = \xi_2$. Since $\xi_1 \geq \xi_2$, it is either $i_1 < i_2$ or $i_1 = i_2$ and $j_1 \leq j_2$. In the first case, the sum of the radii of the j_1-th planet of ring i_1 and the j_2-th satellite of ring i_2 is

$$\begin{aligned} 4(i_1-1)+2+\frac{j_1}{8i_1-5} &\leq 4(i_2-2)+3 \\ &< 4(i_2-1)+2+\frac{2j_2+1}{16i_2-10} \end{aligned}$$

which is the distance of the center of the j_2-th satellite of ring i_2 from the center of the planets. In the second case, the sum of the radii of the j_1-th planet and the j_2-th satellite of ring i_1 is

$$\begin{aligned} 4(i_1-1)+2+\frac{j_1}{8i_1-5} &\leq 4(i_1-1)+2+\frac{2j_1+1}{16i_1-10} \\ &\leq 4(i_2-1)+2+\frac{2j_2+1}{16i_2-10} \end{aligned}$$

which is the distance of the center of the j_2-th satellite of ring i_2 from the center of the planets. Hence, in both cases, the two disks do not overlap.

Finally, consider a pair of nodes $v_s^{\xi_1}$ and $v_s^{\xi_2}$ not connected with an edge in G_{4d^2}. The corresponding disks are satellites; we will show that no two satellites overlap. Consider first the j_1-th satellite of ring i_1 and the j_2-th satellite of ring $i_2 > i_1$. Then, the difference of the distances of the centers of the two satelites from the center of the planets is

$$4(i_2-1)+2+\frac{2j_2+1}{16i_2-10}-4(i_1-1)-2-\frac{2j_1+1}{16i_1-10} >$$
$$4(i_2-i_1)-\frac{2j_1+1}{16i_1-10} > 4(i_2-i_1)-2 \geq 2$$

meaning that the two satellites (which have unit radii) do not overlap. In order to show that no two satellites of the same ring overlap, we will show that no satellite crosses the boundaries of its sector. Consider the j-th satellite of ring i. The angle defined by the bisection of the sector i containing the satellite and one of its boundaries is $\frac{\pi}{8i-4}$. Let δ be the distance of the center of the satellite from the boundary of sector j. Observe that the distance of the center of the satellite from the center of the planets is greater than $4i-2$. Hence,

$$\begin{aligned} \delta &> (4i-2)\sin\frac{\pi}{8i-4} \\ &\geq (4i-2)\frac{2}{\pi}\frac{\pi}{8i-4} \\ &= 1, \end{aligned}$$

where the last inequality follows since for any $x \in [0, \pi/4]$, it is $\sin x \geq \frac{2}{\pi}x$.

□

This completes the proof of the Theorem. □

The lower bound stated by Theorem 39 obviously does not hold when the disk representation is given as part of the input. In this case, a simple deterministic on-line algorithm accepting unit disks (whenever this was possible) and rejecting all other disks would be optimal for sequences of disks produced by the adversary of Theorem 39.

The following theorem gives a lower bound on the competitiveness of on-line independent set algorithms when the disk representation is given as part of the input and essentially states that algorithm Classify is optimal within a constant factor.

Theorem 41. *Any randomized on-line algorithm for computing independent sets in σ-bounded disk graphs with n nodes is $\Omega(\min\{n, \log \sigma\})$-competitive against oblivious adversaries.*

Proof. Given $\sigma \geq 1$ and integer $n \geq 2$, we will construct an adversary similar to the one used in the proof of Theorem 39 with the difference that, in each level, the adversary reveals the representation of the disks to the algorithm. So, an algorithm applied on a sequence generated by the adversary may use the disk representation in order to make its random choices.

Let $\kappa = \lfloor \min\{n/2, 1 + \log \sigma\} \rfloor$. The adversary generates a sequence $\mathcal{D}$ of disks in κ levels $0, 1, ..., \kappa - 1$, each level having two disks. All disks are centered at points of a line so we use only one coordinate for locating their centers. First, the adversary presents two disks of radii σ at level 0 with centers at points $c_l^0 = -\sigma$ and $c_r^0 = \sigma$. For $i = 1, ..., \kappa - 1$, the two disks of level i have radii $\sigma/2^i$ and are presented after the disks of level $i-1$. Let c_l^{i-1}, $c_r^{i-1} = c_l^{i-1} + \sigma/2^{i-2}$ be the coordinates of the centers of the disks of level $i-1$. The adversary tosses a coin in order to locate the disks of level i. On heads, the disks are centered at points $c_l^i = c_l^{i-1} - \sigma/2^i$ and $c_r^i = c_l^{i-1} + \sigma/2^i$; on tails, the disks are centered at points $c_r^i = c_l^{i-1} + \sigma/2^{i-2} - \sigma/2^i$ and $c_l^i = c_l^{i-1} + \sigma/2^{i-2} + \sigma/2^i$, respectively. The construction is depicted in Figure 4.4.

Observe that the disks of the same level are non-overlapping and, for $i = 0, ..., \kappa - 2$, all disks generated after the disks of level i overlap with exactly one of the two disks of level i. The intersection graph of $\mathcal{D}$ is the graph G_κ used in the proof of Theorem 39 for $\kappa = \lfloor \min\{n/2, 1 + \log \sigma\} \rfloor$. Observe that, in each level, both disks have identical radii, they overlap with the same set of disks of smaller levels, and all disks that will appear in the next levels are selected equiprobably to be overlapping with exactly one of them. So, no extra information is actually obtained by the disk representation in each level and the proof completes similarly to the proof of Theorem 39 to obtain that no algorithm can be better than $\frac{\kappa+1}{2}$-competitive. $\square$

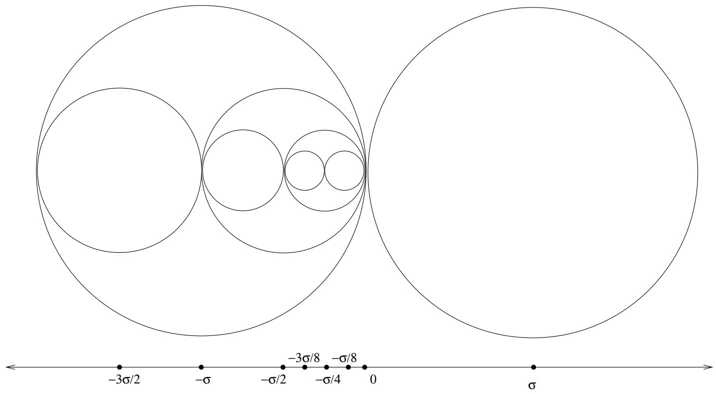

Figure 4.4: The construction of the sequence $\mathcal{D}$ of disks generated by the adversary for the proof of Theorem 6.

4.3 Independent sets in unit disk graphs

In this section, we present new upper and lower bounds on the competitiveness of on-line randomized independent set algorithms for unit disk graphs.

We first present algorithm Filter, an on-line randomized algorithm for computing independent sets in unit disk graphs. We show that the algorithm is $\frac{8\sqrt{3}}{\pi} \approx 4.41$-competitive against oblivious adversaries.

At the beginning, algorithm Filter selects α and β uniformly at random from the intervals $[0, 4)$ and $[0, 2\sqrt{3})$, respectively.

[1]

When a new disk centered at point (x, y) appears, the algorithm does the following: If there are integers κ, λ such that the point $(x + \alpha, y + \beta)$ has distance less than 1 from the point with coordinates $(4\kappa + 2(\lambda \bmod 2), 2\lambda\sqrt{3})$, then Filter executes algorithm First-Fit, else it ignores the disk.

When a new disk centered at point (x, y) appears, the algorithm does the following: If there are integers κ, λ such that the point $(x + \alpha, y + \beta)$ has distance less than 1 from the point with coordinates $(4\kappa + 2(\lambda \bmod 2), 2\lambda\sqrt{3})$, then Filter executes algorithm First-Fit, else it ignores the disk.

[1] Note that we define algorithm Filter in this way for simplifying its description. To make the algorithm more practical, we can modify it so that it selects β uniformly at random in the interval $[0, \delta)$ for all real numbers $\delta > 2\sqrt{3}$ and uses points with coordinates $(4\kappa + 2(\lambda \bmod 2), \lambda\delta)$ instead of $(4\kappa + 2(\lambda \bmod 2), 2\lambda\sqrt{3})$. Then, following the same analysis as the one of Theorem ??, we can show that algorithm Filter achieves a competitive ratio of $\frac{4\delta}{\pi}$ against oblivious adversaries.

Theorem 42. *Algorithm* Filter *is* $\frac{8\sqrt{3}}{\pi}$*-competitive against oblivious adversaries.*

Proof. Consider the application of algorithm Filter on a sequence $\mathcal{D}$ of disks of unit radius. Let $\mathcal{D}'$ denote the (random) subsequence of $\mathcal{D}$ consisting of the disks not ignored by the algorithm. We denote by $A(\mathcal{D})$ the maximum independent set of a sequence $\mathcal{D}$ and by $B(\mathcal{D})$ the set of disks accepted by the algorithm.

We first show that the probability that a disk is not ignored by the algorithm is $\frac{\pi}{8\sqrt{3}}$. Consider a disk D with center at point (x, y) and the rectangle defined by the diagonal points (x, y) and $(x + 4, y + 2\sqrt{3})$. Also, consider the unit disks containing the points at distance less than 1 from points with coordinates $(4\kappa + 2(\lambda \bmod 2), 2\lambda\sqrt{3})$ for integer κ and λ, and observe that the total area of the intersection of these disks with the rectangle equals the area of a disk with radius 1 (see Figure 4.5. Since point $(x + \alpha, y + \beta)$ is uniformly distributed within the rectangle, the probability that the disk D is not ignored by algorithm Filter is equal to the area of a disk of radius 1 over the area of the rectangle, i.e., $\frac{\pi}{8\sqrt{3}}$.

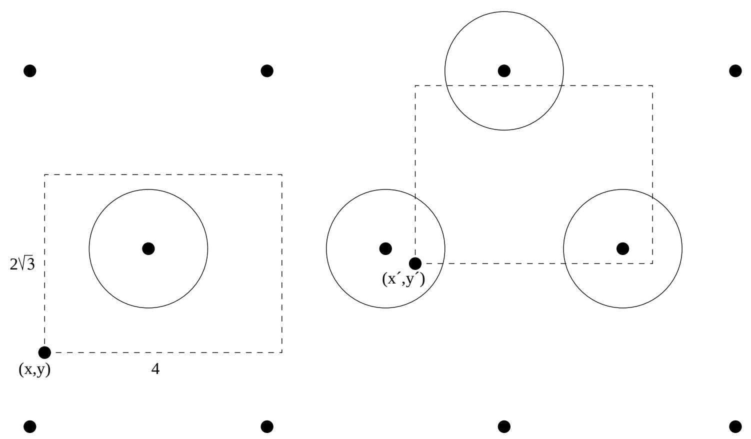

Figure 4.5: Disk centers (x, y) and (x', y') and the rectangles where points $(x + \alpha, y + \beta)$ and $(x' + \alpha, y' + \beta)$ are uniformly distributed.

Now consider the maximum independent set $A(\mathcal{D})$ and let $A'(\mathcal{D})$ be the (random) subset of $A(\mathcal{D})$ consisting of the disks of $A(\mathcal{D})$ not ignored by algorithm Filter. Clearly, $A'(\mathcal{D})$ is an independent set for the set of disks $\mathcal{D}'$, thus, it is $|A(\mathcal{D}')| \geq |A'(\mathcal{D})|$. By linearity of expectation, we obtain that

$|A'(\mathcal{D})| = \frac{\pi}{8\sqrt{3}}|A(\mathcal{D})|$ meaning that

$$E[|A(\mathcal{D}')|] \geq \frac{\pi}{8\sqrt{3}}|A(\mathcal{D})| \tag{4.4}$$

We now observe that each connected component of the intersection graph defined by the disks in $\mathcal{D}'$ is a clique. In particular, consider the two points O_1 with coordinates $(4\kappa_1 + 2(\lambda_1 \bmod 2), 2\lambda_1\sqrt{3})$ and O_2 with coordinates $(4\kappa_2 + 2(\lambda_2 \bmod 2), 2\lambda_2\sqrt{3})$ such that either $\kappa_1 \neq \kappa_2$ or $\lambda_1 \neq \lambda_2$ and three disks D_1, D_2, and D_3 centered at points C_1, C_2, and C_3 with coordinates (x_1, y_1), (x_2, y_2), and (x_3, y_3), respectively. Also, denote by C'_1, C'_2, and C'_3 the points with coordinates $(x_1+\alpha, y_1+\beta)$, $(x_2+\alpha, y_2+\beta)$, and $(x_3+\alpha, y_3+\beta)$, respectively. Assume that points C_1 and C_3 have distance smaller than 1 from point O_1, and point C_2 has distance smaller than 1 from point O_2. We will show that disks D_1 and D_3 overlap while disks D_1 and D_2 are non-overlapping. Clearly, it is $|C_1C_3| = |C'_1C'_3|$ and by triangle inequality, we obtain that $|C_1C_3| \leq |C'_1O_1| + |O_1C'_2| < 2$. Hence, disks D_1 and D_3 overlap. Now, it can be easily verified that if either $\kappa_1 \neq \kappa_2$ or $\lambda_1 \neq \lambda_2$, it is $|O_1O_2| \geq 4$. By the triangle inequality, we have that $|O_1C'_1| + |C'_1C'_2| + |C'_2O_2| \geq 4$. Clearly, $|C'_1C'_2| = |C_1C_2|$ and since $|O_1C'_1| < 1$ and $|C'_2O_2| < 1$, it is also $|C_1C_2| > 2$ meaning that disks D_1 and D_2 do not overlap.

Now, since each connected component of the intersection graph of $\mathcal{D}'$ is a clique, the maximum independent set in the neighborhood of a disk has size at most 1. So, any disk accepted by algorithm Filter may block at most one disk in $A(\mathcal{D}')$. Hence, for the subsequence $\mathcal{D}'$ of the disks not ignored by algorithm Filter, it is $B(\mathcal{D}') \geq |A(\mathcal{D}')|$ implying that $E[B(\mathcal{D}')] \geq E[|A(\mathcal{D}')|]$. Using (4.4), we obtain that the competitive ratio of algorithm Filter is

$$\frac{|A(\mathcal{D})|}{E[B(\mathcal{D}')]} \leq \frac{8\sqrt{3}}{\pi}.$$

□

By adapting the lower bound construction of Section 4.2.2 to the case of unit disk graphs, we obtain the following statement.

Theorem 43. *No on-line (randomized) algorithm for computing independent sets in unit disk graphs can be better than* 3*-competitive against oblivious adversaries if the disk representation is not given. Even if the disk representation is given, then no on-line (randomized) algorithm can be better than* 2.5*-competitive against oblivious adversaries.*

Proof. The proof of the first part is similar to the proof of Theorem 39. We construct an adversary which generates the graph G_5 and show that G_5 is a unit disk graph. A disk representation of G_5 with unit disks is depicted in Figure 4.6.

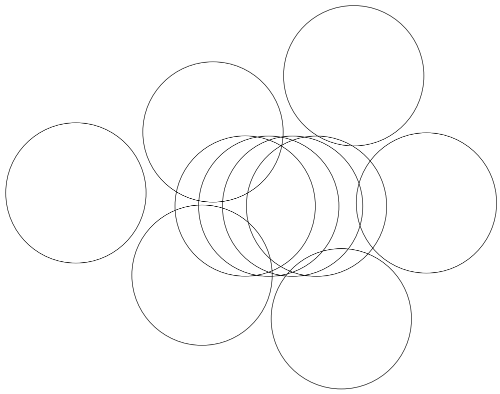

Figure 4.6: The construction of the sequences of disks generated by the adversary for the proof of the first part of Theorem 43.

The proof of the second part is similar to the proof of Theorem 41 and must guarantee that the unit disks in each level but the last one are positioned in such a way (i.e., symmetrically) that no algorithm can gain anything from the representation. We can show that there is a disk representation of G_4 with these properties. An example is depicted in Figure 4.7.

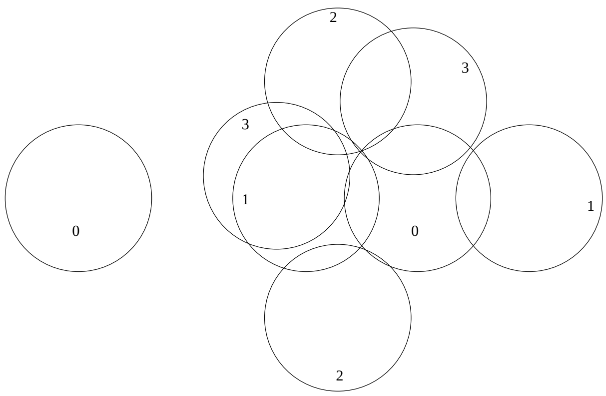

Figure 4.7: An example of the construction of the sequences of disks generated by the adversary for the proof of the second part of Theorem 43.

First, the adversary presents two unit disks at level 0 centered at points $(-2,0)$ and $(2,0)$. Next, it tosses a coin. On:

- HEADS, it presents two unit disks at level 1 centered at points $(-3.3,0)$ and $(-0.7,0)$. Then, it tosses a coin. On:
  - HEADS, it presents two unit disks at level 2 centered at points $(-3,1.7)$ and $(-3,-1.7)$. Then, it tosses a coin. On:
    - HEADS, it presents two unit disks at level 3 centered at points $(-3.7,0.2)$ and $(-2.1,1.5)$. Then, it stops.
    - TAILS, it presents two unit disks at level 3 centered at points $(-3.7,-0.2)$ and $(-2.1,-1.5)$. Then, it stops.
  - TAILS, it presents two unit disks at level 2 centered at points $(-1,1.7)$ and $(-1,-1.7)$. Then, it tosses a coin. On:
    - HEADS, it presents two unit disks at level 3 centered at points $(-0.3,0.2)$ and $(-1.9,1.5)$. Then, it stops.
    - TAILS, it presents two unit disks at level 3 centered at points $(-0.3,-0.2)$ and $(-1.9,-1.5)$. Then, it stops.
- TAILS, it presents two unit disks at level 1 centered at points $(0.7,0)$ and $(3.3,0)$. Then, it tosses a coin. On:
  - HEADS, it presents two unit disks at level 2 centered at points $(1,1.7)$ and $(1,-1.7)$. Then, it tosses a coin. On:
    - HEADS, it presents two unit disks at level 3 centered at points $(0.3,0.2)$ and $(1.9,1.5)$. Then, it stops.
    - TAILS, it presents two unit disks at level 3 centered at points $(0.3,-0.2)$ and $(1.9,-1.5)$. Then, it stops.
  - TAILS, it presents two unit disks at level 2 centered at points $(3,1.7)$ and $(3,-1.7)$. Then, it tosses a coin. On:
    - HEADS, it presents two unit disks at level 3 centered at points $(3.7,0.2)$ and $(2.1,1.5)$. Then, it stops.
    - TAILS, it presents two unit disks at level 3 centered at points $(3.7,-0.2)$ and $(2.1,-1.5)$. Then, it stops.

We can easily see that in every case (depending on the outcome of the coin tosses) the resulting set of disks is a disk representation for the graph G_4. An example is depicted in Figure 4.7. $\square$

4.4 An upper bound for on-line coloring

In this section we present an on-line coloring algorithm for disk graphs which does not require the disk representation. It achieves competitive ratio $O(\min\{\log n, \log \sigma\})$ for coloring σ-bounded sequences of n disks matching the best known upper bound for the case where the disk representation is given. The algorithm is a combination of algorithm First-Fit and algorithm Layered which is presented in the following.

The algorithm Layered classifies the disks into layers and applies algorithm First-Fit to each layer separately, using a different set of colors in each layer. Layers are numbered with integers $1, 2, ...$ and a disk is classified into the smallest layer possible under the constraint that it cannot be classified into a layer if it overlaps with at least 16 mutually non-overlapping disks belonging to this layer.

Lemma 44. *For any σ-bounded sequence of disks, the number of layers constructed by algorithm* Layered *is at most $1 + \log \sigma$.*

Proof. Consider the application of the algorithm on a σ-bounded sequence of disks. Consider a disk D_i belonging to layer $i > 1$ and let R be the radius of D_i. We can show that there is a disk of radius at most $R/2$ in layer $i-1$. In particular, let S be a set of 16 mutually non-overlapping disks of layer $i-1$ overlapping with D_i. Using an argument similar to the one used in the proof of Lemma 34, we can show that at least one of the disks in S has radius at most $R/2$, otherwise, either the disks in S would not be mutually non-overlapping or at least one of them would not overlap with D_i.

We will first show that there is a disk of radius at most $R/2$ in layer $i-1$. Consider a disk D'_i with the same center as D_i and of radius $2R$ and let $D^1_{i-1}, ..., D^{17}_{i-1}$ be the 17 mutually non-overlapping disks of layer $i-1$ overlapping with D_i. Assume that disks $D^1_{i-1}, ..., D^{17}_{i-1}$ all have radius at least $R/2$. Then, the intersection of each of them with D'_i has area at least $\pi R^2/4$ while the area of D'_i is $4\pi R^2$. Since the 17 disks are non-overlapping, so are the areas intersecting with D'_i and it should be $17 \times \pi R^2/4 \leq 4\pi R^2$, a contradiction. Hence, we have proved that at least one of the disks $D^1_{i-1}, ..., D^{17}_{i-1}$ has radius smaller than $R/2$.

Let $j > 1$ be the highest layer containing disks and let D_j be a disk belonging to layer j. This means that one of the disks of layer $j-1$ (say D_{j-1}) that D_j overlaps with has radius at most $R/2$. Using the same argument, we obtain that a disk D_{j-2} of layer $j-2$ overlapping with D_{j-1} has radius at most $R/4$, and, eventually, a disk D_1 of the first layer has radius r at most $R/2^{j-1}$. Hence, $\sigma \geq R/r \geq 2^{j-1}$ which gives that the number of layers j is $j \leq 1 + \log \sigma$. □

Theorem 45. *The algorithm* Layered *is* $O(\log \sigma)$*-competitive when applied to* σ*-bounded sequences of disks.*

Proof. Consider the application of algorithm Layered on a σ-bounded sequence of disks. Let j be the layer where the maximum number of colors has been used. Let α be the highest color used in layer j and let D_j be the disk colored with color α. Then, this disk overlaps with at least $\alpha - 1$ disks of layer j appeared prior to it. By the definition of the algorithm, the number of mutually non-overlapping disks of layer j overlapping with D_j is at most 15. This implies that the optimal algorithm should use at least $\alpha/15$ colors for coloring the disks of the sequence while, by Lemma 44, algorithm Layered uses at most $\alpha(\log \sigma + 1)$ colors, hence, it is at most $15(\log \sigma + 1)$-competitive. □

We now combine algorithms First-Fit and Layered using a technique proposed in [19, 20] to obtain a better result. We use two separate sets of colors for algorithms First-Fit and Layered. When a new disk D_i is presented we run algorithm First-Fit on D_i together with those disks colored by First-Fit. Similarly, we execute Layered. Then, we compare the results of these two algorithms and color D_i with the algorithm that has used fewer colors up to that point (including the color used for disk D_i). The total number of colors used is at most the sum of the number of colors used by both methods. Note that at any time of the execution of the combined algorithm, the number of colors used by First-Fit and the number of colors used by Layered differ by at most one. Assume that $n < \sigma$. The number of colors used by First-Fit is at most $O(\log n)$ times the optimal number of colors. The number of colors used by Layered is at most one more than that of First-Fit. So, the total number of colors is at most $O(\log n)$ times the optimal number of colors. A similar argument holds in the case where $n \geq \sigma$. We obtain the following theorem.

Theorem 46. *There exists an* $O(\min\{\log n, \log \sigma\}$*-competitive algorithm for on-line coloring a* σ*-bounded disk graph with* n *nodes.*

Chapter 5

Conclusions - Future Research

In this PhD Thesis, we have presented techniques for the efficient solution of two fundamental graph-theoretic problems: maximum independent set and graph coloring with the further objective of their application to the efficient frequency allocation and call control in wireless networks.

We have studied the on–line version of both problems using competitive analysis. We have considered cellular, planar and arbitrary network topologies. For the frequency allocation problem in cellular networks, we improved the best known competitive ratio which had been proved to be at least 3. Using competitive analysis, we proved that the competitive ratio of the greedy algorithm is at least 2.429 and at most 2.5.

For the call control problem in cellular networks, we presented algorithm p-random, a randomized algorithm that uses randomness proportional to the size of the network in wireless networks of reuse distance 2 and proved that it achieves a competitive ratio which is better than the corresponding deterministic. In particular, we proved that algorithm p-random obtains a competitive ratio against oblivious adversaries between 2.469 and 2.651. Our analysis can extend to arbitrary networks. Furthermore, using Yao's Minimax Principle we proved lower bounds of 1.857 and 2.086 on the competitiveness of randomized call control algorithms for wireless networks with cellular, arbitrary and planar interference graphs, respectively.

We significantly improved these results giving a series of simple randomized algorithms which obtain competitive ratios less than 3, work for networks that support arbitrarily many frequencies and, use only either a constant number of random bits or a weak random source. The best upper bound on competitiveness we proved is $7/3$.

For cellular networks of reuse distance $k > 2$, we presented simple randomized on-line call control algorithms with competitive ratios that significantly improve the lower bounds of deterministic algorithms and use only $O(\log k)$ random bits. Furthermore, we proved new lower bounds of $25/12$, $127/60$, and $2,5$ on the competitiveness of on-line call control algorithms in cellular networks of reuse distance $k = 2, 3, 4$, $k = 5$ and $k \geq 6$, respectively.

We have studied the on-line versions of the two fundamental graph-theoretic problems, the maximum independent set problem and the graph coloring problem, for disk graphs, which are graphs resulting from the intersection of disks on the plane. In disk graphs, the maximum independent set and the coloring problems are important since they can be used to model resource allocation problems in wireless communication networks [29]. For the maximum independent set problem, we examined if the use of randomness can help in the improvement of the competitive ratios of on-line algorithms. We proved that, in general, the use of randomness does not help against oblivious adversaries even when the disk representation is given as part of the input, i.e., we constructed sequences of disks for which no (possibly randomized) on-line algorithm can have a competitive ratio better than $\Omega(n)$. When the disk representation is not given as part of the input, we proved a lower bound of $\Omega(\min\{n, \sigma^2\})$ on the competitiveness of on-line algorithms in σ-bounded disk graphs with n vertices which implies that algorithm First-Fit is optimal within a constant factor. For σ-bounded disk graphs when the disk representation is given as part of the input, we presented randomized algorithms with competitive ratios almost logarithmic in σ and proved that these algorithms can be optimal. For unit disk graphs, we presented a randomized algorithm with competitive ratio equal to 4.41 (which is less than the lower bound of 5 on the competitiveness of deterministic algorithms). We also proved lower bounds of 2.5 and 3 on the competitiveness of randomized algorithms for unit disk graphs. For the coloring problem, we showed how the best known upper bound of $O(\min\{\log n, \log \sigma\})$ for σ-bounded sequences of n disks can be achieved even if the disk representation is not given as part of the input.

Our work reveals some interesting open problems for both the frequency allocation and the call control problems.

For the frequency allocation problem, the most interesting open problem is whether there exists a 2-competitive deterministic algorithm. Furthermore, the use of randomness could give improved results for the problem. To the best of our knowledge, randomized frequency allocation algorithms have not been studied in cellular networks, neither for the static nor for the on-line version. In this case, the only known lower bound is the one of 4/3 (which is only implied by Bartal *et al.* in [8]).

For the call control problem, the basic open problem is to close the gap between the upper and lower bound on the competitiveness of the relevant algorithms. The most interesting version of the problem concerns randomized on-line algorithms in wireless cellular networks with small reuse distance. We also presented algorithms based on the ““classify and randomly select”” paradigm using new colorings for the interference graph. These algorithms use a small number of random bits, and obtain small competitive ratios against oblivious adversaries even in the case of networks that support arbitrarily many frequencies.

Furthermore, our approach naturally leads to the definition of the following coloring problem. To our knowledge, this problem has not been studied before.

> GRAPH COLORING BY SEPARATED CLIQUES
> INSTANCE: A graph $G = (V, E)$ and a positive integer K.
> QUESTION: Is there an assignment of at most K colors to the nodes of G such that each connected component of the subgraph of G induced by the nodes colored with the same color is a clique?

Efficient solutions to this problem (upper bounds on the number of colors and approximability results for its optimization version), even for special classes of graphs, may give simple and efficient competitive on-line algorithms based on the “classify and randomly select” paradigm in more general network topologies than those discussed in this PhD Thesis.

The results for the independent set extend to the more general problem where we are given $w \geq 1$ colors and the objective is to accept the maximum number of disks which can be properly colored with at most w colors (clearly, for $w = 1$, this is the independent set problem). Algorithms Classify, Guess, and Filter can be easily modified to solve this problem with the same competitiveness bounds we proved for the independent set problem. Out techniques can also be used for other classes of intersection graphs, e.g., rectangle intersection graphs.

The most interesting open problem related to the independent set problem is perhaps to close the gap on the competitiveness of (randomized) on-line algorithms in unit disk graphs. It would be very interesting even to find an algorithm with competitive ratio smaller than 5 which does not require the disk representation. For the coloring problem, there is still a large gap (in terms of σ) between the competitiveness of algorithm Layered (or algorithms with similar competitive ratio that use the disk representation) and the known lower bounds.

Bibliography

[1] B. Awerbuch, Y. Azar, A. Fiat, S. Leonardi, and A. Rosen. Competitive On–line Call Admission in Optical Networks. In *Proceedings of the 4th Annual European Symposium on Algorithms (ESA '96)*, LNCS 1136, Springer, pp. 431–444, 1996.

[2] B. Awerbuch, Y. Azar, A. Fiat, S. Leonardi, and A. Rosen. On-line competitive algorithms for call admission in optical networks. *Algorithmica*, Vol. 31(1), pp. 29-43, 2001.

[3] B. Awerbuch, Y. Bartal, A. Fiat, A. Rosen. Competitive Non–Preemptive Call Control. In *Proccedings of the 5th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA '94)*, pp. 312–320, 1994.

[4] P.K. Agarwal, M. Overmars, and M. Sharir. Computing maximally separated sets in the plane and independent sets in the intersection graph of unit disks. In *Proc. of the 15th Annual ACM/SIAM Symposium on Discrete Algorithms (SODA '04)*, 2004, to appear.

[5] B. Baker. Approximation algorithms for $NP-$complete problems on planar graphs. *Journal of the ACM*, 41(1):153-180, 1994.

[6] H. Breu. Algorithmic aspects of constrained unit disk graphs. *PhD Dissertation*, University of British Columbia, 1996.

[7] A. Borodin and R. El-Yaniv. Online computation and competitive analysis. *Cambridge University Press*, 1998.

[8] Y. Bartal, A. Fiat, and S. Leonardi. Lower Bounds for On–line Graph Problems with Applications to On–line Circuit and Optical Routing. In *Proc. of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, pp. 531–540, 1996.

[9] T.M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *Journal of Algorithms*, 46, pp. 178-189, 2003.

[10] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165-177, 1990.

[11] I. Caragiannis, C. Kaklamanis, and E. Papaioannou. On–line Call Control in Cellular Networks. *Foundations of Mobile Computing* (satellite workshop of FST&TCS '99), 1999.

[12] I. Caragiannis, C. Kaklamanis, and E. Papaioannou. Efficient On–Line Communication in Cellular Networks. In *Proc. of the 12th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA '00)*, pp. 46–53, 2000.

[13] I. Caragiannis, C. Kaklamanis, and E. Papaioannou. Competitive Analysis of On–line Randomized Call Control in Cellular Networks. In *Proc. of the 15th International Parallel and Distributed Processing Symposium (IPDPS '01)*, 2001.

[14] I. Caragiannis, C. Kaklamanis, and E. Papaioannou. Randomized Call Control in Sparse Wireless Cellular Networks. In *Proc. of the 8th International Conference on Advances in Communications and Control (COMCON 01)*, pp. 73-82, 2001.

[15] I. Caragiannis, C. Kaklamanis, and E. Papaioannou. Efficient On–Line Frequency Allocation and Call Control in Cellular Networks. *Theory of Computing Systems*, Vol. 35, pp. 521-543, 2002.

[16] I. Caragiannis, C. Kaklamanis, and E. Papaioannou. Simple on-line algorithms for call control in cellular networks. In *Proc. of the 1st Workshop on Approximation and On-line Algorithms (WAOA '03)*, LNCS 2909, Springer, pp. 67-80, 2003.

[17] I. Caragiannis, A. Fishkin, C. Kaklamanis, E. Papaioannou. On-line algorithms for disk graphs. In *Proc. of the 29th International Symposium on Mathematical Foundations of Computer Science (MFCS '04)*, LNCS 3153, Springer, pp. 215-226, 2004.

[18] D. Dimitrijević and J. Vučetić. Design and performance analysis of algorithms for channel allocation in cellular networks. *IEEE Trnsactions on Vehicular Technology*, 42(4): 526-534, 1993

[19] T. Erlebach and J. Fiala. On-line coloring of geometric intersection graphs. *Computational Geometry: Theory and Applications*, 9(1-2), pp. 3-24, 2002.

[20] T. Erlebach and J. Fiala. Independence and coloring problems on intersection graphs of disks. In *Approximation and On-line Algorithms*, LNCS. Springer Verlag, 2004. To appear.

[21] T. Erlebach and K. Jansen. The Maximum Edge-Disjoint Paths Problem in Bidirected Trees. *SIAM Journal on Discrete Mathematics*, Vol. 14(3), pp. 326-355, 2001.

[22] T. Erlebach, K. Jansen, and E. Seidel. Polynomial-time approximation schemes for geometric graphs. In *Proc. of the 12th Annual ACM/SIAM Symposium on Discrete Algorithms (SODA '01)*, pp. 671-679, 2001.

[23] A.V. Fishkin. Disk graphs: a short survey. In *Proc. of the 1st International Workshop on Approximation and On-line Algorithms (WAOA '03)*, LNCS 2909, Springer, pp. 260-264, 2003.

[24] J. Fiala, A.V. Fishkin, and F.V. Fomin. Off-line and on-line distance constrained labeling of graphs. In *Proc. of the 9th Annual European Symposium on Algorithms (ESA '01)*, LNCS 2161, pp. 464-475, 2001.

[25] D. Fotakis, S. Nikoletseas, V. Papadopoulou and P. Spirakis. NP-completeness Results and Efficient Approximations for Radiocoloring in Planar Graphs. In *Proceedings of the 25th International Symposium on Mathematical Foundations of Computer Science (MFCS 2000)*, LNCS 1893, Springer, pp. 363-372, 2000.

[26] A. Gräf. Coloring and recognizing special graph classes. Musikinformatik und Medientechnik Bericht 20/95, Johannes Gutenberg, Universität Mainz, 1995.

[27] A. Gräf, M. Stumpf, and G. Weissenfels. On coloring unit disk graphs. *Algorithmica*, 20(3), pp. 277-293, 1998.

[28] A. Gyárfás and J. Lehel. On-line and first fit colorings of graphs. *Journal of Graph Theory*, 12(2), pp. 217-227, 1988.

[29] W.K. Hale. Frequency Assignment: Theory and Applications. In *Proceedings of the IEEE*, 68(12), pp. 1497–1514, 1980.

[30] M.M. Halldórsson. Approximating discrete collections via local improvements. In *Proc. of the 6th Annual ACM/SIAM Symposium on Discrete Algorithms (SODA '95)*, pp. 160-169, 1995.

[31] P. Hliněný and J. Kratochvíl. Representing graphs by disks and balls. *Discrete Mathematics*, 229(1-3), pp. 101-124, 2001.

[32] D. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Applied Mathematics*, 6, pp. 243-254, 1983.

[33] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130-136, 1985.

[34] H.B. Hunt III, M.V. Marathe, V. Radhakrishnan, S.S. Ravi, D.J. Rosenkrantz, and R.E. Stearns. NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs. *Journal of Algorithms*, 26(2), pp. 238-274, 1998.

[35] S. Irani. Coloring inductive graphs on-line. *Algorithmica*, 11, pp. 53-72, 1994.

[36] J. Janssen and K. Kilakos. Optimal multicolouring algorithms with limitied recolouring. Submitted for publication, April 1995.

[37] J. Janssen, K. Kilakos, and O. Marcotte. Ficed preference frequency allocation for cellular telephone systems. IEEE Trans. Veh. Techn. 48(2): 533-541, 1999.

[38] J. Janssen, D. Krizanc, L. Narayanan, and S. Shende. Distributed On–Line Frequency Assignment in Cellular Networks. In *Proceedings of 15th Annual Symposium on Theoretical Aspects of Computer Science (STACS '98)*, LNCS 1373, Springer, pp. 3–13, 1998.

[39] J. Janssen, D. Krizanc, L. Narayanan, and S. Shende. Distributed On–Line Frequency Assignment in Cellular Networks. *Journal of Algorithms*, Vol. 36(2), pp. 119-151, 2000.

[40] T. Kahwa and N. Georganas. A hybrid channel assignment scheme in large-scale cellular-structured mobile communication systems. *IEEE Transactions on Communications.* 4:432-438, 1978.

[41] S. Kim and S. L. Kim. A two-phasealgorithm for frequency assignment in cellular mobile systems. *IEEE Transactions on Vehicular Technology*, 1994.

[42] S. Leonardi, A. Marchetti–Spaccamela, A. Prescuitti, and A. Rosen. On–line Randomized Call–Control Revisited. In *Proccedings of the 9th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA '98)*, pp. 323–332, 1998.

[43] L. Lovasz, M. Saks, and W. Trotter. An online graph coloring algorithm with sub-linear performance ratio. *Discrete Math*, 75:319-325, 1989.

[44] E. Malesińska. Graph theoretical models for frequency assignment problems. *PhD Thesis*, Technical University of Berlin, 1997.

[45] T. Matsui. Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs. In *Discrete and Computational Geometry*, LNCS 1763, Springer, pp. 194-200, 2000.

[46] M.V. Marathe, H. Breu, H.B. Hunt III, S.S. Ravi, and D.J. Rosenkrantz. Simple hueristics for unit disk graphs. *Networks*, 25, pp. 59-68, 1995.

[47] R. Motwani and B. Raghavan. Randomized Algorithms. *Cambridge University Press*, 1995.

[48] C. McDiarmid and B. Reed. Channel Assignment and Weighted Coloring. Manuscript, 1997.

[49] L. Narayanan and S. Shende. Static Frequency Assignment in Cellular Networks. In *Proceedings of the 5th International Colloquium on Structural Information and Communication Complexity (SIROCCO '97)*, pp. 215–227, 1997.

[50] L. Narayanan and Y. Tang. Worst-Case Analysis of a Dynamic Channel Assignment Strategy. Discrete Applied Mathematics 140(1-3): pp. 115-141, 2004. Preliminary version in *Proceedings of the 4th International Workshop on Discrete Alforithms and Methods for Mobile Computing and Communications (DIALM 2000)*, 2000.

[51] R. Peeters. On coloring j-unit sphere graphs. Technical report, Dept. of Economics, Tilburg University, 1991.

[52] G. Pantziou, G. Pentaris, and P. Spirakis. Competitive Call Control in Mobile Networks. *Theory of Computing Systems*, Vol. 35(6), pp. 625-639, 2002. Preliminary version in *Proceedings of International Sym-*

posium on Algorithms and Computation (ISAAC '97), LNCS 1350, Springer, pp. 404–413, 1997.

[53] P. Raymond. Performance analysis of cellular netowrks. *IEEE Transaction on Communications*. 39(12):1787-1793, 1991.

[54] D. Sleator and R.E. Tarjan. Amortized Efficiency of List Update and Paging Rules. *Communications of Association of Computing Machinery* 28, pp. 202–208, 1985.

[55] L. Trevisan. Non–Approximability Results for Optimization Problems on Bounded Degree Instances. In *Proc. of the 33rd Annual ACM Symposium on Theory of Computing (STOC '01)*, pp. 453–461, 2001.

[56] Y.-T. Tsai, Y.-L. Lin, and F.R. Hsu. The on-line first-fit algorithm for radio frequency assignment problems. *Information Processing Letters*, 84, pp. 195-199, 2002.

[57] S. Vishwanathan. Randomized online graph coloring. *Journal of Algorithms*, 13:657-669, 1992.

[58] D. Wang and Y.-S. Kuo. A study on two geometric location problems. *Information Processing Letters*, 28:281-286, 1988.

[59] P.-J. Wan and L. Liu. Maximal Throughput in Wavelength-Routed Optical Networks. Multichannel Optical Networks: Theory and Practice, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, AMS, Vol. 46, pp. 15-26, 1998.

[60] W. Wang and C. Rushforth. An adaptive local-search algorithms for the channel-assignment problem. Technical Report, August 1995.

[61] A. C. Yao. Probabilistic Computations: Towards a Unified Measure of Complexity. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science (FOCS '77)*, pp. 222–227, 1977.