MANAGEMENT SCIENCE
Articles in Advance, pp. 1–20
ISSN 0025-1909 (print) | ISSN 1526-5501 (online)

http://dx.doi.org/10.1287/mnsc.2015.2221
© 2015 INFORMS

Constrained Resource Assignments: Fast Algorithms and Applications in Wireless Networks

André Berger
Department of Quantitative Economics, Maastricht University, 6200 MD Maastricht, Netherlands,
a.berger@maastrichtuniversity.nl

James Gross
School of Electrical Engineering, KTH Royal Institute of Technology, S100 44 Stockholm, Sweden,
james.gross@ee.kth.se

Tobias Harks
Institute of Mathematics, University of Augsburg, 86135 Augsburg, Germany,
tobias.harks@math.uni-augsburg.de

Simon Tenbusch
Institute for Operations Research and Management GmbH, 52076 Aachen, Germany,
simon.tenbusch@inform-software.com

Resource assignment problems occur in a vast variety of applications, from scheduling problems over image recognition to communication networks. Often these problems can be modeled by a maximum weight matching problem in (bipartite) graphs or generalizations thereof, and efficient and practical algorithms are known for these problems. Although in some of the applications an assignment of the resources may be needed only once, in many of these applications, the assignment has to be computed more often for different scenarios. In that case it is often essential that the assignments can be computed very fast. Moreover, implementing different assignments in different scenarios may come with a certain cost for the reconfiguration of the system. In this paper, we consider the problem of determining optimal assignments sequentially over a given time horizon, where consecutive assignments are coupled by constraints that control the cost of reconfiguration. We develop fast approximation and online algorithms for this problem with provable approximation guarantees and competitive ratios. Moreover, we present an extensive computational study about the applicability of our model and our algorithms in the context of orthogonal frequency division multiple access (OFDMA) wireless networks, finding a significant performance improvement for the total bandwidth of the system using our algorithms. For this application (the downlink of an OFDMA wireless cell) , the run time of matching algorithms is extremely important, having an acceptable range of a few milliseconds only. For the considered realistic instances, our algorithms perform extremely well: the solution quality is, on average, within a factor of 0.8–0.9 of optimal off-line solutions, and the running times are at most 5 ms per phase even in the worst case. Thus, our algorithms are well suited to be applied in the context of OFDMA systems.

Data, as supplemental material, are available at http://dx.doi.org/10.1287/mnsc.2015.2221.

Keywords: mobile networks; frequency allocation; matchings; information systems; enabling technologies; analysis of algorithms; approximation algorithms
History: Received March 15, 2013; accepted March 23, 2015, by Teck-Hua Ho, optimization. Published online in *Articles in Advance*.

1. Introduction

Resource assignment problems play a key role in many practical applications. Whenever a set of resources needs to be matched to a set of demands, the goal is to find the most profitable or least costly assignment of the resources to the demands. Assuming that each resource might have a different profit or cost for each demand and each resource can be assigned to at most one demand, usually this problem can be modeled by a maximum weight matching problem in (bipartite) graphs or generalizations thereof. Applications come from a wide range of areas, including scheduling (Höhn et al. 2011), image recognition (Kim and Kak 1991), telecommunications (Goudreau et al. 2000, Urgaonkar and Neely 2009, Zhang and Yang 2004, Zhao et al. 2008), and game theory (Gusfield and Irving 1989, Knuth 1976).

Resource allocation problems in telecommunication networks have been addressed, for example, in the context of switching (Goudreau et al. 2000) and wavelength division multiplexing in optical networks (Zhang and Yang 2004). They are also omnipresent in wireless networks such as in orthogonal frequency division multiple access (OFDMA) networks and cognitive (wireless) networks (Urgaonkar and Neely 2009, Zhao et al. 2008). Here, the set of resources often models the set of available wireless channels, whereas mobile clients represent the demands. The corresponding profit for assigning a channel to a client depends on the channel states, which in turn depend on several factors such as the movement of the client, its distance to the transmitter, and interference. Because the state of a wireless channel changes relatively fast (within tens of milliseconds in general), efficient resource allocation algorithms are of interest, such as those used for the bipartite weighted matching problem (Kim et al. 2001, Urgaonkar and Neely 2009, Yin and Liu 2000, Zhao et al. 2008).

In addition to the necessity of solving the resource allocation problem fast, it is often necessary to compute the assignments repeatedly over time. Because the profits or costs of assigning resources to the demands may change over time (for various system specific reasons), the corresponding optimal assignment may also change. If the system is switched to a new assignment, reconfiguration costs can occur that have a negative impact on the overall system performance. Examples of such reconfiguration costs are setup costs for machines in the context of scheduling (Höhn et al. 2011) or control information in wireless networks (Gross et al. 2006, Henttonen et al. 2008). Typically, this reconfiguration cost grows with the difference (e.g., the number of changed edges) between the assignments in consecutive phases.

Motivated by such reconfiguration costs, in this paper we consider two models that take these costs into account. The first model is based on the k-*constrained* bipartite matching problem, where the objective is to compute a maximum weight (perfect) matching such that no more than k edges are modified with respect to a given initial (perfect) matching. This problem arises as a first natural extension of the classical bipartite matching problem by assuming that the system operates within two phases (corresponding to changed edge weights) and the reconfiguration costs are controlled by imposing a budget constraint of k new edges. We also consider the more general case, where the edge weights may vary over several phases. Because edge weights for the different consecutive phases are usually not known beforehand (for example, due to the unknown future channel states in a wireless network), this leads to a natural *online* variant of the k-*constrained* matching problem. We develop efficient and competitive online algorithms for the case of multiple phases in which assignments for the phases have to be found such that every two consecutive assignments respect the budget constraint, and edge weights of future phases are not known beforehand.

Whereas the k-*constrained* matching problem imposes a hard budget constraint on the number of changed edges, as a second model we also consider the case of *elastic* reconfiguration costs, where possibly all edges can be changed, but the weight of the new matching linearly decreases with the number of changed edges. For this model we also develop efficient and competitive online algorithms for the case of multiple phases.

1.1. Related Work

Starting from Kuhn's (1955) seminal contribution on the Hungarian method, there has been a tremendous amount of work addressing the problem of designing efficient algorithms for different variants of matching problems (for corresponding surveys, see Galil 1986, Korte and Vygen 2000, Schrijver 2003).

Matching problems with coupling constraints have not been considered much in the literature. Most closely related to this work from an algorithmic perspective are bicriteria formulations of the matching problem. In that sense, the reconfiguration costs can be modeled as a second weight function. Earlier work on the bicriteria problem focused on the construction of the Pareto curve (Papadimitriou and Yannakakis 2000) or on the budgeted version of the problem. Recently, a polynomial time approximation scheme (PTAS) was developed for the closely related budgeted matching problem with general weights and costs (Berger et al. 2011). Although the problems in Berger et al. (2011) are related to our work, this PTAS cannot be applied to our models, since it has running times that are far from being of practical relevance, and since it cannot be used to find budgeted *perfect* matchings. The corresponding budgeted *perfect* matching problem is NP-hard as well, and no approximation algorithms for it are known. Other related work on these problems has been carried out by Papadimitriou and Yannakakis (1982), who developed very general approaches for approximation algorithms for problems with a constant number of objectives, based on the construction of ϵ-approximate Pareto curves.

In the context of telecommunication networks, it has been shown that the control information can become a significant drawback in the downlink of OFDMA systems (Gross et al. 2006, Henttonen et al. 2008). To reduce the signaling overhead, different techniques have been studied. A quadratic optimization model that maximizes net throughput was proposed by Gross et al. (2006). From an online perspective, approaches for resource allocation and channel assignments in wireless networks have been considered, for example, by Buchbinder et al. (2012) and Fu et al. (2006) in the context of power allocation and data scheduling for data transmission using dynamic programming. Finally, Midran et al. (2010) consider online assignment algorithms for resource allocation in OFDMA systems taking into consideration the utility of terminals as a recursive function over time.

1.2. Our Contribution and Organization

After introducing the model and necessary notation in §2, we derive a fast approximation algorithm (Algorithm 1) for the k-constrained bipartite matching problem in §3. We prove that the solution computed by our algorithm guarantees at least 50% of the maximum possible weight. In §4, we formally introduce the online k-constrained bipartite matching problem, where the goal is to determine matchings sequentially over time and in an online fashion; that is, edge weights of future phases are not revealed to the algorithm. We first show an upper bound of $(k-1)/n$ on the competitive ratio of any deterministic online algorithm and then introduce an online algorithm for this problem (that is based on the previously introduced algorithm for two phases) having a competitive ratio with an (almost) matching lower bound of $(\lfloor k/2\rfloor)/n$. To evaluate the solution quality of our algorithm for real-world instances, we also derive a compact linear integer programming (IP) formulation for the off-line optimization problem.

In §5, we introduce an online matching problem with elastic reconfiguration costs. For this variant, we develop an efficient online algorithm that has a competitive ratio of 1/9 for all bipartite graphs with at least three nodes on one partition. We further show that this algorithm is the best possible by deriving an upper bound of 1/9 for any deterministic online algorithm. Also for this variant, we derive a compact integer linear programming formulation for the off-line optimization problem.

In §6, we numerically evaluate the presented algorithms in the context of the downlink of an OFDMA cell. We evaluate running times of our algorithms and compare solution quality (for both variants) with upper bounds on the corresponding off-line optimal solution (which are based on solving the linear relaxation of the above-mentioned integer programs) as well as with comparison schemes from literature. It turns out that for realistic instances, the algorithms' performances greatly exceed their theoretically proven approximation guarantees. On the test set representing different mobility and interference scenarios of an OFDMA cell, the quality of our solutions is, on average, within a factor of 0.8–0.9 of the optimal solutions, and they outperform various comparison schemes significantly with respect to different performance metrics (including a measure for quality of service as well as one for fairness). Although there is a mild decrease in performance over the optimal off-line solution, the run times of our algorithms are quite fast, taking only up to 5 ms per phase on the instances. Thus, our algorithms are well suited to be applied in the context of OFDMA systems, where computation times must lie in the range of milliseconds.

2. The Model

We start this section with formally defining the *bipartite matching problem with reconfiguration costs* and introducing the necessary notation. For some integer $n \geq 1$ we consider the balanced complete bipartite graph $G_n = K_{n,n}$ with n vertices in each partite set. The vertex set of G_n consists of two disjoint sets $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$, each of cardinality n, and its edge set $E_n = \{u_i v_j\colon 1 \leq i, j \leq n\}$ consists of all edges between U and V. Let M denote a perfect matching in G_n, and let $PM(G_n)$ denote the set of perfect matchings of G_n. The mapping $w\colon E_n \to \mathbb{R}_0^{+}$[1] is called the weight function, and $w(M)$ is the weight of the matching M. In a *sequential* bipartite matching problem we are given a sequence of edge weights $\sigma = (w^1, \ldots, w^T)$, where $w^t\colon E_n \to \mathbb{R}_0^+$, $t = 1, \ldots, T$, $T \in \mathbb{N}$. Here, T denotes the number of time slots. To model reconfiguration costs, we introduce for any two consecutive matchings (M^{t-1}, M^t) a cost or penalty function $c\colon PM(G_n) \times PM(G_n) \to \mathbb{R}$. The cost value $c(M^{t-1}, M^t)$ scales down the achievable weight due to reconfiguring M^{t-1} to obtain M^t. Given an initial matching M^0, the overall objective is to calculate perfect matchings M^t, $t = 1, \ldots, T$, such that the total *net weight*

$$\sum_{t=1}^{T} w^t(M^t) \cdot c(M^{t-1}, M^t)$$

is maximized. Note that every two consecutive matchings M^{t-1} and M^t are coupled by the cost function $c(M^{t-1}, M^t)$. In the remainder of this paper, we will be concerned with two important variants of the cost function.

2.1. Budget-Constrained Matching

The first variant that we address is the k-constrained bipartite matching problem, where we are given a budget constraint on the number of changed edges. Formally, we are given an integer parameter $k \geq 0$, and define the cost function $c(M^{t-1}, M^t)$ as

$$c(M^{t-1}, M^t) = \begin{cases} 1 & \text{if } |M^t \cap M^{t-1}| \geq n-k, \\ -\infty & \text{otherwise.} \end{cases}$$

This type of cost function represents a hard budget constraint requiring that at most k edges can be changed per time slot.

[1] Throughout the paper, we use $\mathbb{R}_0^+$ to denote the set of nonnegative real numbers.

The sequential k-constrained bipartite matching problem is then to find a sequence of perfect matchings $(M^1, \dots, M^T)$ of maximum total weight. Certainly, for any optimal solution, every two consecutive matchings M^{t-1} and M^t will differ by at most k edges; that is, for every $t \in [T]$, M^t will satisfy $|M^t \cap M^{t-1}| \geq n-k$. For ease of presentation we assume that the initial matching satisfies $M^0 = \{u_i v_i\colon 1 \leq i \leq n\}$.

2.2. Elastic Reconfiguration Costs

In the second variant, we assume that for every two consecutive matchings, the weight of the new matching linearly decreases with the number of changed edges. More precisely, suppose we are given an initial matching M^{t-1} and a weight function w^t. Then the reconfiguration cost of the matching M^t is defined as

$$c(M^{t-1}, M^t) = \kappa + (1-\kappa) \cdot \frac{|M^t \cap M^{t-1}|}{n}.$$

Here, $\kappa \in [0,1]$ is a parameter that represents the impact of reconfiguration costs on the obtained weight. For instance, a value of $\kappa = 1$ reduces our problem to a maximum weight perfect matching problem without reconfiguration costs. In contrast, a low value of κ represents high reconfiguration costs, which lead to a lower total net weight.

3. Budget-Constrained Matchings

We first investigate the case of hard budget constraints on the number of changed edges. We call the corresponding problem the k-constrained matching problem, where k refers to the value of the actual budget imposed. Before we study the general case of T time slots, we first study the seemingly easy case of a single time slot only. Our insights for the case of a single time slot will later be used as the main building block of algorithms for the general case. In this problem, we are given an initial matching M^0 and the goal is to compute a matching M of maximum weight that differs from M^0 in at most k edges. This problem arises as a first natural extension of the classical bipartite matching problem by assuming that the system operates within two phases (corresponding to changed edge weights) and the reconfiguration costs are controlled by imposing a strict budget constraint of k new edges. As noted by Berger et al. (2011), despite several efforts over the last decade, the complexity status of this problem is still open; that is, neither a polynomial time algorithm is known nor is the problem known to be NP-hard. In this paper, we tackle the problem by using approximation algorithms. For a maximization problem, a polynomial time algorithm is called an α-approximation for some $\alpha \in [0,1]$ if the algorithm computes a solution of weight at least α times the weight of an optimal solution. We devise a simple and fast $\lfloor k/2 \rfloor / k$-approximation (Algorithm 1) that only needs to execute two maximum weight perfect matching problems on a modified instance.

The main idea of Algorithm 1 is to change the weights of the edges slightly (Step 1) to account for an edge being either in the initial matching or not. Therefore, a *new* edge is penalized by having its weight reduced and is thus less likely to be included in the solution. The weight of such an edge which is not in the matching M^0 is reduced by the average weight of its two incident edges from M^0.

Algorithm 1 (A $\lfloor k/2 \rfloor / k$-Approximation for the k-Constrained Bipartite Matching Problem)

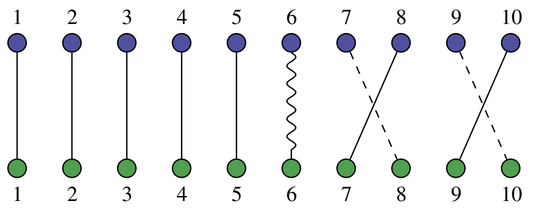
```
Require: A complete bipartite graph
         $G_n = K_{n,n} = (U \cup V, E_n)$ with edge weights
         $w_{ij} \in \mathbb{R}_0^+$, the initial perfect matching
         $M^0 = \{u_i v_i\colon 1 \leq i \leq n\}$ and a parameter
         $k \geq 0$
Ensure: A perfect matching $M$ of $G_n$ with
         $|M \cap M^0| \geq n-k$
 1. set $w(M^0) = \sum_{i=1}^{n} w_{ii}$
 2. for all $1 \leq i, j \leq n$ do
 3.   set $w'_{ij} = w_{ij} + w(M^0)/k - (w_{ii} + w_{jj})/2$
 4. end for
 5. find a maximum weight matching $M_1$ w.r.t. $w'$
      with at most $l := \lfloor k/2 \rfloor$ edges
 6. $I := \varnothing$; $M^0_{ALG} := \varnothing$
 7. for all $1 \leq i \leq n$ do
 8.   if $u_i$ and $v_i$ are not matched by $M_1$ and
        $|M^0_{ALG}| < n-k$ then
 9.     $I := I \cup \{i\}$; $M^0_{ALG} := M^0_{ALG} \cup \{u_i v_i\}$
10.   end if
11. end for
12. compute a maximum weight perfect matching
      $M^1_{ALG}$ w.r.t. $w$ on the subgraph $G[\{u_i, v_i\colon i \notin I\}]$
      of nodes not matched by edges in $M^0_{ALG}$.
13. return $M = M^0_{ALG} \cup M^1_{ALG}$
```

Theorem 1. *Algorithm 1 is a $\lfloor k/2 \rfloor / k$-approximation for the k-constrained bipartite matching problem and runs in $O(kn^3)$.*

Proof. We first show that the algorithm indeed outputs a feasible solution to the k-constrained bipartite matching problem. Let $l := \lfloor k/2 \rfloor$, and let M_1 be the matching computed in Step 5 of the algorithm. Let $I = \{i\colon u_i v_i \in M^0_{ALG}\}$, where M^0_{ALG} is defined in Step 9 of Algorithm 1. By the choice of l, there are at most $2l \leq k$ edges from the initial matching M^0 that are incident with an edge from M_1. Therefore, there are at least $n - 2l \geq n-k$ potential edges that can be added to M^0_{ALG} in Steps 7–11. Since $M^0_{ALG} \subseteq M^0$ we have that $|M \cap M^0| \geq n-k$, and hence, the output M is feasible.

For proving the approximation ratio of the algorithm, we will compare the weight of the returned

Figure 1 **(Color online) An Example of the Different Edge Sets in the Solution Described in the Proof of Theorem 1**



Notes. Here $n = 10$ and $k = 5$. In this example, $M^0 = \{(i,i), i = 1,\ldots,10\}$, and Algorithm 1 computes $M_1 = \{(8,7),(10,9)\}$ in Step 5. Then, we obtain $M^0_{ALG} = \{(i,i), i = 1,\ldots,5\}$, $M_1 = \{(8,7),(10,9)\}$, $M_2 = \{(7,8),(9,10)\}$, and $\tilde{M} = \{(6,6)\}$.

matching M with the weight of another feasible solution $\bar{M}$. For this purpose, we define

$$\tilde{M} = \{u_i v_i \colon i \notin I \text{ and } u_i \text{ and } v_i \text{ are not end points of edges in } M_1\}.$$

In words, $\tilde{M}$ contains those edges from M^0 that, besides edges in M^0_{ALG}, could additionally be used after having computed M_1. Moreover, we let M_2 be an arbitrary matching that, when added to $M^0_{ALG} \cup \tilde{M} \cup M_1$, will yield a perfect matching of G_n. We denote this matching by $\bar{M} = M^0_{ALG} \cup \tilde{M} \cup M_1 \cup M_2$ (see Figure 1 for an illustration).

In the remaining part of the proof we will show that $w(\bar{M}) \geq \lfloor k/2 \rfloor / k w(\textsc{Opt})$. This implies the claimed approximation guarantee using $w(M) = w(M^0_{ALG}) + w(M^1_{ALG}) \geq w(M^0_{ALG}) + w(\tilde{M}) + w(M_1) + w(M_2) = w(\bar{M})$. The above inequality follows since M^1_{ALG} was computed as a maximum weight perfect matching on the same set of nodes that are also spanned by $\tilde{M} \cup M_1 \cup M_2$.

We will now prove the inequality $w(\bar{M}) \geq \lfloor k/2 \rfloor / k w(\textsc{Opt})$ by establishing four claims. In these claims, we use the notation $\textsc{Opt} = \textsc{Opt}_0 \cup \textsc{Opt}_1$ with $\textsc{Opt}_0 \subseteq M^0$ and $|\textsc{Opt}_0| = n - k$ describing a decomposition of $\textsc{Opt}$ into $n-k$ "old" edges and possibly k "new" edges.

Claim 1. $w(\bar{M}) = w'(\tilde{M} \cup M_1 \cup M_2)$.

Claim 2. $w'(M_2 \cup \tilde{M}) \geq 0$.

Claim 3. $w'(M_1) \geq \lfloor k/2 \rfloor / k w'(\textsc{Opt}_1)$.

Claim 4. $w'(\textsc{Opt}_1) = w(\textsc{Opt})$.

From the above claims it follows that

$$\begin{aligned} w(\bar{M}) &\overset{\text{Claim 1}}{=} w'(M_1 \cup M_2 \cup \tilde{M}) = w'(M_1) + w'(M_2 \cup \tilde{M}) \\ &\overset{\text{Claim 2}}{\geq} w'(M_1) \overset{\text{Claim 3}}{\geq} \frac{\lfloor k/2 \rfloor}{k} w'(\textsc{Opt}_1) \\ &\overset{\text{Claim 4}}{=} \frac{\lfloor k/2 \rfloor}{k} w(\textsc{Opt}). \end{aligned}$$

Now we prove the claims.

Proof of Claim 1. By definition of w' we obtain

$$\begin{aligned} &w'(M_1 \cup M_2 \cup \tilde{M}) \\ &\quad = \sum_{u_i v_j \in M_1 \cup M_2 \cup \tilde{M}} \left(w_{ij} + \frac{w(M^0)}{k} - \frac{w_{ii} + w_{jj}}{2} \right) \\ &\quad = w(M_1 \cup M_2 \cup \tilde{M}) + k \cdot \frac{w(M^0)}{k} - \sum_{i \notin I} w_{ii} \\ &\quad = w(M_1 \cup M_2 \cup \tilde{M}) + w(M^0_{ALG}) = w(\bar{M}). \end{aligned}$$

Proof of Claim 2. Since $|M_1 \cup M_2 \cup \tilde{M}| = k$ and $|M_1| \leq \lfloor k/2 \rfloor$, we have that $|M_2 \cup \tilde{M}| \geq k/2$. Therefore, we get that

$$\begin{aligned} w'(M_2 \cup \tilde{M}) &= \sum_{u_i v_j \in M_2 \cup \tilde{M}} \left(w_{ij} + \frac{w(M^0)}{k} - \frac{w_{ii} + w_{jj}}{2} \right) \\ &\geq \frac{k}{2} \cdot \frac{w(M^0)}{k} + \sum_{u_i v_j \in \tilde{M}} 0 - \sum_{u_i v_j \in M_2} \frac{w_{ii} + w_{jj}}{2} \\ &\geq \frac{w(M^0)}{2} - \frac{w(M^0)}{2} = 0, \end{aligned}$$

where the last inequality holds because in the sum $\sum_{u_i v_j \in M_2} (w_{ii} + w_{jj}) 2$ each term w_{ii} can appear only once. To see this, observe that if w_{ii} appeared twice, we would get that $u_i v_j \in M_2$ and $u_{j'} v_i \in M_2$ for some j, j', implying $u_i v_i \in \tilde{M}$, a contradiction.

Proof of Claim 3. Consider an optimal solution $\textsc{Opt}$ and a decomposition $OPT = OPT_0 \cup OPT_1$, where $OPT_0 \subseteq M^0$ and $|OPT_0| = n - k$. Then $|OPT_1| = k$, and the set L of the $\lfloor k/2 \rfloor$ heaviest edges of $\textsc{Opt}_1$ with respect to (w.r.t.) w' (independent on whether some of these weights may be negative) has the property that

$$w'(L) \geq \frac{\lfloor k/2 \rfloor}{k} w'(\textsc{Opt}_1).$$

Because L comprises a feasible solution to the problem solved in Step 5 of the algorithm, we obtain

$$w'(M_1) \geq w'(L) \geq \frac{\lfloor k/2 \rfloor}{k} w'(\textsc{Opt}_1).$$

Proof of Claim 4. Let $I^* = \{i \colon u_i v_i \in \textsc{Opt}_0\}$. We now have that

$$\begin{aligned} w'(\textsc{Opt}_1) &= \sum_{u_i v_j \in \textsc{Opt}_1} \left(w_{ij} + \frac{w(M^0)}{k} - \frac{w_{ii} + w_{jj}}{2} \right) \\ &= w(\textsc{Opt}_1) + k \cdot \frac{w(M^0)}{k} - \sum_{i \notin I^*} w_{ii} \\ &= w(\textsc{Opt}_1) + \sum_{i \in I^*} w_{ii} = w(\textsc{Opt}). \end{aligned}$$

Regarding the claimed running time of the algorithm, note that the problem in Step 5 can be solved by adding two independent sets of size $n-r$ ($1 \leq r \leq l$) and connecting all the vertices of the first set to U and all vertices of the second set to V with edges of weight zero, and then finding a maximum weight perfect matching in the augmented graph. This will return a maximum weight matching in the original graph having exactly r edges. Applying this procedure for all $1 \leq r \leq l$ and choosing the matching of maximum weight will give the desired matching having at most l edges. This also implies that the algorithm can be implemented to run in $O(kn^3)$ time. □

For even k, Algorithm 1 is a 1/2-approximation. If k is odd and small (say $k \leq 1/\epsilon + 1$), then the optimal solution can be found by exhaustive search. On the other hand, if $k > 1/\epsilon + 1$, then $\lfloor k/2 \rfloor / k \geq 1/2 - \epsilon$. This implies that Algorithm 1 can be used to devise a $(1/2 - \epsilon)$-approximation for all k's that run in time $O(n^{1/\epsilon})$. Also note that our algorithm works for complete graphs as well without modification, except that for Steps 5 and 12, a maximum weight perfect matching algorithm for general graphs has to be used.

4. Online Budget-Constrained Matching Problems

In this section, we introduce an online variant of the k-constrained bipartite matching problem that captures the sequential structure of systems that arise in practice.

An instance of the online k-constrained bipartite matching problem consists of a balanced complete bipartite graph $G_n = K_{n,n} = (U \cup V, E_n)$ with n nodes in each partite set. Moreover, we are given a perfect matching M^0 in G_n and a sequence of edge weights $\sigma = (w^1, \ldots, w^T)$, where $w^t\colon E_n \to \mathbb{R}_0^+$, $t = 1, \ldots, T$, $T \in \mathbb{N}$. Here, T denotes the number of time slots. The goal is to sequentially calculate perfect matchings M^t, $t = 1, \ldots, T$ such that the total weight $\sum_{t=1}^{T} w^t(M^t)$ is maximized. We make the following three crucial assumptions: (i) edge weights are revealed in an *online fashion*, that is, edge weights are only revealed for the current time slot and future edge weights are not known; (ii) once a matching is determined, no change of this matching is possible; (iii) every matching M^t may have at most k changes with respect to its predecessor matching M^{t-1}, $t = 1, \ldots, T$. Note that constraint (iii) is equivalent to using the scaled weight function $\sum_{t=1}^{T} c(M^{t-1}, M^t) w^t(M^t)$, with $c(M^{t-1}, M^t) = 1$, if $|M^t \cap M^{t-1}| \geq n-k$ and $-\infty$ otherwise.

Knowing all edge weights in advance, we call the problem of maximizing the total weight subject to the matching constraints the *off-line optimization problem* and denote the off-line optimal solution (and its total weight) by Opt.

4.1. Online Algorithms and Competitive Analysis

For a given sequence of weights $\sigma = (w^1, \ldots, w^T)$ and a sequence of perfect matchings $(M^1, \ldots, M^T)$ produced by an online algorithm, Alg, we denote by $\sigma(\text{Alg})$ the total weight of all perfect matchings in the output sequence. The online algorithm Alg is called (strictly) *c-competitive*, if for all possible sequences σ, $\sigma(\text{Alg})$ is never smaller than c times the total weight of an optimal off-line solution. The *competitive ratio* of Alg is the supremum over all $c \geq 0$ such that Alg is c-competitive; see, for instance, Borodin and El-Yaniv (1998) and Fiat and Woeginger (1998).

We first present an algorithm that achieves a competitive ratio of $\lfloor k/2 \rfloor / n$. The idea of the algorithm is similar to the one used in Algorithm 1. Given edge weights w^t in time slot t and a perfect matching M^{t-1}, we first compute a maximum weight matching w.r.t. w^t having at most $\lfloor k/2 \rfloor$ edges. This matching can be extended to a perfect matching having at most k changes from M^{t-1}; see Algorithm 2 for a formal description. The main difference to Algorithm 1 appears in Step 1 of Algorithm 2, where a maximum weight perfect matching M' w.r.t. w^t is computed (instead of the changed edge weights w'). For the sake of simplicity, we extend M' to an arbitrary perfect matching M^t because the way M' is extended does not change the competitive ratio. Moreover, note that this algorithm does not actually consider the input matching M^{t-1} and returns a feasible matching having the claimed approximation ratio, even when compared to the optimal value of the assignment problem without any constraints.

Algorithm 2 (A $\lfloor k/2 \rfloor / n$-Competitive Algorithm for Online k-Constrained Bipartite Matching)

```
Require: A complete bipartite graph
         $G_n = K_{n,n} = (U \cup V, E_n)$ with edge weights
         $w_{ij}^t \in \mathbb{R}_0^+$, the previous perfect matching
         $M^{t-1} = \{u_i v_i : 1 \leq i \leq n\}$ and a parameter
         $k \geq 0$
Ensure: A perfect matching $M^t$ of $G_n$ with
        $|M^t \cap M^{t-1}| \geq n-k$
1. find a maximum weight matching $M'$ w.r.t. $w^t$
      with at most $l := \lfloor k/2 \rfloor$ edges
2. extend $M'$ to an arbitrary perfect matching $M^t$
```

We complement this result by showing that no deterministic online algorithm can achieve a competitive ratio better than $(k-1)/n$, thus matching our bound up to a factor of 2.

Theorem 2. *The competitive ratio of the above online algorithm is at least $\lfloor k/2 \rfloor / n$, where n is the size of the balanced complete bipartite input graph G_n. Moreover, the competitive ratio of any deterministic online algorithm is at most $(k-1)/n$, even when all weights are restricted to be in the set $\{0, 1\}$.*

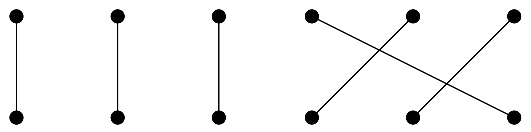


Figure 2 **Lower Bound Construction for $n = 3$**

Notes. The left perfect matching has been computed by an arbitrary deterministic online algorithm in phase $T-1$. The right perfect matching is optimal for phase T. The weights in phase T are 1 for visible edges and 0 for all other edges in G_3.

Proof. Let $\sigma = (w^1, \dots, w^T)$ be arbitrary and consider the two solutions, Alg, which is produced by the above algorithm, and Opt, the optimal solution for the corresponding off-line problem with weight sequence σ. Let Opt^t and Alg^t denote the solutions of Opt and Alg in time slot t, respectively. We can argue that the $\lfloor k/2 \rfloor$ heaviest edges of Opt^t comprise a feasible solution to the problem that Alg solves in time slot t (see Step 1 of Algorithm 2). Clearly, the weight of the $\lfloor k/2 \rfloor$ heaviest edges of Opt^t sum up to at least $(\lfloor k/2 \rfloor / n) w^t(\text{Opt}^t)$. Hence, for every $1 \le t \le T$ we have that $w^t(\text{Alg}^t) \ge (\lfloor k/2 \rfloor / n)\, w^t(\text{Opt}^t)$, and the claimed competitive ratio follows as we sum up this inequality over all time slots.

For proving the upper bound, we construct an instance of the k-constrained online matching problem with $T \ge \lfloor n/k \rfloor + 1$ time slots as follows. The initial matching M^0 is any arbitrary perfect matching in G_n. We specify $\sigma = (w^1, \dots, w^T)$ as follows. All weights of the first $T-1$ time slots remain zero; i.e., $w^t_{ij} = 0$ for all $1 \le t \le T-1$ and all $1 \le i, j \le n$. Let Alg be an arbitrary deterministic online algorithm that determines the matching M^{T-1} in time slot $T-1$. By relabeling indices we can assume $(i, i) \in M^{T-1}$ for $i = 1, \dots, n$. Given M^{T-1}, the online adversary determines the edge weights for time slot T as follows. We define $w^T_{i,\,i+1} = 1$, for $i = 1, \dots, n-1$, and $w^T_{n,\,1} = 1$, $w^T_{ij} = 0$ otherwise. Clearly, Alg achieves for the first $T-1$ time slots a total weight of 0. In the last phase, Alg can add at most $k-1$ edges of weight 1, because obtaining $k-1$ edges of weight 1 in phase T requires adding at least one new edge of weight 0; see Figure 2 for an illustration for the case $n = 3$. Thus, the total weight for all T phases is at most $(k-1)/n$. The optimal matching (anticipating the high weight in the last time slot) will be able to successively add k new edges (that have weight 1 in the last phase) in every time slot (possibly less in the last time slot if $k \nmid n$). Since there are $T \ge \lfloor n/k \rfloor + 1$ time slots, it can achieve an overall weight of n, and thus $\sigma(\text{Alg}) \le ((k-1)/n)\sigma(\text{Opt})$. □

Note that the above algorithm and Algorithm 1 differ only in the weight functions that are used, and that it is only the modification of the original weights that enables us to achieve a $(\lfloor k/2 \rfloor)/k$-approximation. We can also combine the algorithm from Theorem 2 and Algorithm 1 to obtain an online algorithm with a competitive ratio of $(\lfloor k/2 \rfloor)/n$ that at the same time provides a $(\lfloor k/2 \rfloor)/k$-approximation to the optimal solution for each time slot.

Corollary 1. *The online algorithm that in each time slot chooses from the solutions of the algorithm from Theorem 2 and of Algorithm 1 the one with higher weight has a competitive ratio of at least $(\lfloor k/2 \rfloor)/n$ and provides a $(\lfloor k/2 \rfloor)/k$-approximation for the k-constrained matching problem in each time step.*

4.2. The Off-Line Problem: An Integer Linear Programming Formulation

Whereas Theorem 2 provides an almost optimal lower bound on the competitive ratio achievable by any deterministic online algorithm, the actual competitive ratio $((\lfloor k/2 \rfloor)/n)$ is a *worst-case* bound. For real-world instances, this bound may be far too pessimistic. To evaluate the performance of our online algorithms for real-world instances, we need to solve the corresponding off-line problem. Recall that assuming the weights of all T frames are known, we have to determine T matchings that maximize the total net weight over all T frames while allowing no more than k modifications on the assignments from frame to frame. We will first present a very natural *nonlinear* integer programming formulation. We use the following notation for parameters and variables:

- $w^t_{ij} \in \mathbb{R}^+$: weight of edge ij during phase t ($i, j \in [n]$, $t \in [T]$);
- $x^t_{ij} \in \{0, 1\}$: binary variable indicating use of edge ij during phase t ($i, j \in [n]$, $t \in [T]$).

The following (assignment) constraints are used:

$$\sum_j x^t_{ij} = 1 \quad \text{for all } i \in [n],\ t \in [T], \tag{1}$$

$$\sum_i x^t_{ij} = 1 \quad \text{for all } j \in [n],\ t \in [T]. \tag{2}$$

Given a nonnegative integer parameter $k \le n$, two consecutive matchings are constrained to differ in at most k edges, or, said differently, at least $n-k$ edges must be kept from the matching in a previous phase. An edge ij is kept from phase $t-1$ to phase t if and only if $x^{t-1}_{ij} \cdot x^t_{ij} = 1$. The objective is to maximize the total weight of all edges over all phases. For the off-line optimization problem we hence obtain the following formulation with quadratic constraints:

$$\max \ \sum_t \sum_{i,j} w^t_{ij} \cdot x^t_{ij} \tag{3}$$

$$\text{s.t.} \ \sum_{i,j} x^{t-1}_{ij} x^t_{ij} \ge n-k \quad \text{for all } 2 \le t \le T. \tag{4}$$

In general, the above constraints define a nonconvex boundary of the feasible region, and thus the formulation cannot be solved using standard techniques. To obtain a linear mixed integer formulation, we introduce for every $i, j \in [n]$ and $2 \le t \le T$ the additional binary variables y_{ij}^t defined as

$$y_{ij}^t = \begin{cases} 1 & \text{if edge } ij \text{ is used in phases } t-1 \text{ and } t, \\ 0 & \text{otherwise.} \end{cases}$$

Based on these new variables, we replace (4) by the constraint

$$\sum_{i,j} y_{ij}^t \ge n - k \quad \text{for all } 2 \le t \le T, \tag{5}$$

and we add the constraints

$$y_{ij}^t \le x_{ij}^{t-1} \quad \text{for all } i, j \in [n],\ 2 \le t \le T, \tag{6}$$

$$y_{ij}^t \le x_{ij}^t \quad \text{for all } i, j \in [n],\ 2 \le t \le T \tag{7}$$

to make sure that $y_{ij}^t = 0$ if either $x_{ij}^{t-1} = 0$ or $x_{ij}^t = 0$. This ensures that $y_{ij}^t = 1$ only if edge ij is in both matchings, x^{t-1} and x^t. Therefore, any sequence of feasible matchings $x^1, \dots, x^T$ can be extended to a feasible solution of (5)–(7). Thus, we obtain the following proposition.

Proposition 3. *The integer linear program defined by* (1)–(3) *and* (5)–(7) *is a correct formulation for the budget-constrained matching problem for multiple phases.*

The above integer linear program (or any relaxation thereof) can be solved (for instance, using CPLEX) and yields an upper bound on the objective value of the off-line problem.

5. Online Matchings with Elastic Reconfiguration Costs

As in the previous section, an instance of the online bipartite matching problem with elastic reconfiguration costs consists of a balanced complete bipartite graph $G_n = K_{n,n} = (V \cup W, E_n)$ together with an initial perfect matching M^0 in G_n and a sequence of edge weights $\sigma = (w^1, \dots, w^T)$, where $w^t\colon E_n \to \mathbb{R}_0^+$, $t = 1, \dots, T$, $T \in \mathbb{N}$. The goal is to sequentially calculate perfect matchings M^t, $t = 1, \dots, T$, such that the total net weight $\sum_{t=1}^T w^t(M^t)(\kappa + (1-\kappa)\cdot(|M^{t-1} \cap M^t|)/n)$ is maximized. In this variant, there is a parameter $\kappa \in [0, 1]$ that captures the trade-off between the amount of reconfiguration and the obtained net weight.

We impose the assumptions that edge weights are revealed in an online fashion and, once a matching is determined, no change of this matching is possible. In contrast to the pervious online version assuming a hard budget constraint, now two consecutive matchings may have an arbitrary number of different matching edges.

5.1. Online Algorithms and Competitive Analysis

We now present an online algorithm having a *constant* competitive ratio of 1/9 for arbitrary instances with $n \ge 3$. Note that for $n = 1$ there is nothing to do, and for $n = 2$ one can show that no deterministic online algorithm can have a competitive ratio better than κ, whereas the online algorithm that always computes a maximum weight perfect matching actually achieves a competitive ratio of κ.

For the general case $n \ge 3$, our online algorithm works as follows. At the beginning of every phase, we compute for every $k = 1, \dots, n$ a k-constrained matching using Algorithm 1 and then select the best solution. The time complexity of this algorithm is $O(n^4)$. It turns out that for worst-case instances, the optimal k is equal to $2 \cdot \lfloor n/3 \rfloor$, giving a competitive ratio of precisely 1/9. We further prove that the algorithm is in some sense *best possible* by proving that no deterministic online algorithm can have a competitive ratio above 1/9.

Theorem 4. *For $n \ge 3$, the competitive ratio of the above online algorithm is at least* 1/9, *and it runs in* $O(n^4)$. *Moreover, no deterministic online algorithm can have a competitive ratio above* 1/9.

Proof. We first prove the upper bound. Let $n = 3$ and $T = 2$. The initial matching M^0 is any arbitrary perfect matching in G_3. We specify $\sigma = (w^1, w^2)$ as follows. All weights of the first time slot remain zero, i.e., $w_{ij}^1 = 0$, for all $1 \le i,\ j \le 3$. Let Alg be an arbitrary deterministic online algorithm that determines the matching M^1 in time slot 1. By relabeling indices, we can assume $(i, i) \in M^1$ for $i = 1, \dots, 3$. Given M^1, the online adversary determines the edge weights for time slot $T = 2$ as follows. We define $w_{i,i+1}^2 = 1$, for $i = 1, 2$, and $w_{3,1}^2 = 1$, and $w_{ij}^2 = 0$ otherwise (see again Figure 2 for the construction). Alg achieves for the first time slot a net weight of 0. If Alg picks 1 edge of weight 1 in the last phase, we obtain $\text{Alg}(\sigma) = 1 \cdot (\kappa + (1-\kappa)\cdot(3-2)/3) = \kappa + (1-\kappa)/3$. If Alg picks at least two edges of weight 1 in the last phase, we obtain $\text{Alg}(\sigma) \le 3 \cdot (\kappa + (1-\kappa)\cdot 0) = 3 \cdot \kappa$. The optimal solution requires no reconfiguration and achieves a net weight of 3. For $\kappa = 0$, we thus obtain the upper bound of 1/9.

Now, we prove the lower bound for *arbitrary* $n \ge 3$. Let $\sigma = (w^1, \dots, w^T)$ be an arbitrary sequence, and let $(O^1, \dots, O^T)$ denote an optimal solution and $(M^1, \dots, M^T)$ denote the solution of the online algorithm. Let OPT^i denote the corresponding net weight in slot i, and let ALG^i denote the corresponding net weight of the online algorithm. By the definition of Algorithm 1 for $k = 2\lfloor n/3 \rfloor$, the obtained net weight is larger than or equal to than taking the $\lfloor n/3 \rfloor$ heaviest edges with respect to a maximum perfect matching (maximizing w^i). Moreover, the solution returned by

Algorithm 1 (for $k = 2\lfloor n/3 \rfloor$) results in at most $2\lfloor n/3 \rfloor$ changed edges. Thus, it follows that

$$ALG^i \geq w^i(M^i) \cdot \left(\kappa + \frac{(1-\kappa)\cdot(n - 2\cdot\lfloor n/3 \rfloor)}{n}\right)$$
$$\geq w^i(O^i) \cdot \left(\kappa + \frac{(1-\kappa)\cdot(n - 2\cdot\lfloor n/3 \rfloor)}{n}\right) \cdot \frac{\lfloor n/3 \rfloor}{n}.$$

We now need the following technical lemma.

Lemma 1. *Let $\kappa \in [0,1]$. Then, for all $n \in \mathbb{N}$ with $n \geq 3$, the following inequality holds*:

$$\left(\kappa + \frac{(1-\kappa)\cdot(n - 2\cdot\lfloor n/3 \rfloor)}{n}\right) \cdot \frac{\lfloor n/3 \rfloor}{n} \geq \frac{1}{9} + \frac{4}{45}\cdot\kappa.$$

Proof of Lemma 1. Writing $n \equiv r \mod 3$, for a remainder $r \in \{0,1,2\}$, we have to consider three cases.

1. $r = 0$: We can write $n = q \cdot 3$ for some $q \in \mathbb{N}$. We obtain

$$\left(\kappa + \frac{(1-\kappa)\cdot(n - 2\cdot\lfloor n/3 \rfloor)}{n}\right) \cdot \frac{\lfloor n/3 \rfloor}{n}$$
$$= \frac{\kappa}{3} + (1-\kappa)\cdot\frac{(3q-2q)q}{9q^2} = \frac{1+2\kappa}{9}.$$

2. $r = 1$: We can write $n = q \cdot 3 + 1$ for some $q \in \mathbb{N}$. We obtain

$$\left(\kappa + \frac{(1-\kappa)\cdot(n - 2\cdot\lfloor n/3 \rfloor)}{n}\right) \cdot \frac{\lfloor n/3 \rfloor}{n}$$
$$= \frac{\kappa q}{3q+1} + (1-\kappa)\cdot\frac{(3q+1-2q)q}{(3q+1)^2}$$
$$\geq \frac{\kappa}{4} + \frac{1-\kappa}{9} = \frac{1}{9} + \frac{5\kappa}{36}.$$

For the last inequality we used $q^2 + q \geq q^2 + (7/9)\cdot q + 1/9$ for all $q \in \mathbb{N}$.

3. $r = 2$: We can write $n = q \cdot 3 + 2$ for some $q \in \mathbb{N}$. We obtain

$$\left(\kappa + \frac{(1-\kappa)\cdot(n - 2\cdot\lfloor n/3 \rfloor)}{n}\right) \cdot \frac{\lfloor n/3 \rfloor}{n}$$
$$= \frac{\kappa q}{3q+2} + (1-\kappa)\cdot\frac{(3q+2-2q)q}{(3q+2)^2}$$
$$\geq \frac{\kappa}{5} + \frac{1-\kappa}{9} = \frac{1}{9} + \frac{4\kappa}{45}.$$

For the last inequality we used $q^2 + 2q \geq q^2 + (4/3)\cdot q + 4/9$ for all $q \in \mathbb{N}$. □

The theorem now follows by the above lemma. □

5.2. The Off-Line Problem: An Integer Linear Programming Formulation

Again, as for the online k-constrained matching problem, we need to compute the off-line optimum to compare the performance of the above online algorithms to a theoretically possible upper bound. To compute the off-line optimum, we consider the problem of finding consecutive perfect matchings in complete bipartite graphs with n nodes in each partite set over T phases. We use the following notation:

- $w_{ij}^t \in \mathbb{R}^+$: weight of edge ij during phase t ($i, j \in [n]$, $t \in [T]$);
- $x_{ij}^t \in \{0,1\}$: binary variable indicating use of edge ij during phase t ($i, j \in [n]$, $t \in [T]$).

The following assignment constraints are used:

$$\sum_j x_{ij}^t = 1 \quad \text{for all } i \in [n],\ t \in [T]; \tag{8}$$

$$\sum_i x_{ij}^t = 1 \quad \text{for all } j \in [n],\ t \in [T]. \tag{9}$$

According to the multiphase nature of the problem and the fact that we consider variable signaling costs, given a sequence $(x^t)_{t \in [T]}$ of feasible integral perfect matchings, we obtain the following objective function (the vector x^0 describes the initial matching M^0):

$$f((x^t)_{t\in[T]}) = \sum_{t\geq 1}\left(\left(\sum_{i,j} w_{ij}^t \cdot x_{ij}^t\right)\left(\kappa + \frac{1-\kappa}{n}\sum_{i,j} x_{ij}^{t-1} x_{ij}^t\right)\right). \tag{10}$$

Note that again the above mixed-integer problem formulation is nonlinear. We turn it into a linear formulation by adding the following variables:

- $y_{ij}^t \in \{0,1\}$: binary variable indicating whether edge ij is both in x^{t-1} and in x^t (i.e., $y_{ij}^t = x_{ij}^{t-1} \cdot x_{ij}^t$);
- $y^t \in \mathbb{Z}^+$: number of edges kept from x^{t-1} to x^t;
- $z_{ij}^t \in \mathbb{Z}^+$: equals $x_{ij}^t \cdot y^t$.

Then, the previously nonlinear objective can be turned into a linear one by observing that $\sum_{i,j} x_{ij}^{t-1} x_{ij}^t = y^t$:

$$f((x^t)_{t\in[T]}) = \sum_{t\geq 1}\left(\kappa \cdot \left(\sum_{i,j} w_{ij}^t \cdot x_{ij}^t\right) + \frac{1-\kappa}{n}\cdot\left(\sum_{i,j} w_{ij}^t z_{ij}^t\right)\right). \tag{11}$$

Furthermore, we add the following constraints:

$$y_{ij}^t \leq x_{ij}^t \quad \text{for all } i, j \in [n],\ t \geq 2, \tag{12}$$

$$y_{ij}^t \leq x_{ij}^{t-1} \quad \text{for all } i, j \in [n],\ t \geq 2, \tag{13}$$

$$y^t = \sum_{i,j} y_{ij}^t \quad \text{for all } t \geq 2, \tag{14}$$

$$z_{ij}^t \leq x_{ij}^t \cdot n \quad \text{for all } i, j \in [n],\ t \geq 2, \tag{15}$$

$$z_{ij}^t \leq y^t \quad \text{for all } i, j \in [n],\ t \geq 2. \tag{16}$$

Similarly as in §4.2, constraints (12)–(14) ensure that y^t equals the number of edges kept from x^{t-1} to x^t. Because (11) is a maximization problem, the variables z_{ij}^t will be set to y^t for edges ij that are in the matching x^t, and to 0 for edges for which $x_{ij}^t = 0$. Therefore, in any optimal solution, $z_{ij}^t = x_{ij}^t \cdot y^t$ for all i, j, and t. Hence we obtain the following proposition.

Proposition 5. *The integer linear program defined by (8), (9), and (11)–(16) is a correct formulation for the matching problem with elastic reconfiguration costs.*

6. Computational Study: Resource Assignments in OFDMA Wireless Networks

In this section we evaluate the proposed algorithms for the budget-constrained case and the case of elastic reconfiguration costs. We consider these algorithms in the context of wireless systems, more precisely in the context of the so-called downlink (i.e., the transmission direction from the base station to the terminals) of an OFDMA system. This is the standard transmission technology for example in upcoming fourth-generation cellular networks. In the following, we first introduce the system model and parameterization of the evaluation study. Then we discuss the results for the budget-constrained case and the case of elastic reconfiguration costs regarding two different system scenarios.

6.1. System Overview

OFDMA systems are characterized by a set of n parallel communication channels referred to as subcarriers. These are used to transmit data simultaneously from one point in the system—usually the base station of a cellular network—to multiple different clients (referred to as terminals in the following). OFDMA systems typically operate in a slotted fashion; i.e., time is split into frames of length T_f. We focus in the following purely on the downlink transmission direction and assume that the entire duration of each frame can be used for it. Each frame of duration T_f is furthermore subdivided into S digital symbols, which ultimately convey the information. Hence, with n subcarriers and S symbols, one downlink frame can transport a total of $S \cdot n$ symbols from the base station to the terminals. Note that in general symbols can represent different amounts of bits (as discussed in §6.2).

The base station and terminals are connected physically by the wireless channel. These channels are well known for their unreliable transmission quality, which results from a randomly varying channel gain between transmitter and receiver (with several magnitudes of variations in the gain over tens of milliseconds in common transmission environments). In OFDMA systems, it is well known that the quality of the wireless channels (i.e., the subcarriers) varies over time and frequency. Thus, per downlink phase, different subcarriers each feature a different quality to some distinct terminal. This results in a varying amount of bits that can be transmitted on different subcarriers and/or at different downlink frames. Denote by w_{ij}^t the amount of bits that can be transmitted on subcarrier j to terminal i during downlink frame t. This bit amount varies randomly for different subcarriers, for different terminals, and for different downlink frames. However, the base station of an OFDMA system tracks via feedback loops the state of the wireless subcarriers. Depending on the scenario, this feedback is fast enough to provide an accurate estimate of the next downlink phase. For most urban wireless communication settings, where the movement speed of the objects in the environment is low to medium, this assumption is appropriate (Dahlmann et al. 2008). Therefore, we consider in the following that the base station has perfect channel knowledge prior to the upcoming downlink frame regarding each terminal/subcarrier pair. If instead the feedback is not fast enough, stronger error correction coding needs to be employed, which reduces the amount of bits w_{ij}^t that can be transmitted per terminal/subcarrier pair.

Based on this channel state knowledge, the base station optimizes the allocation of subcarriers to terminals for the upcoming frame. Different objective functions have been discussed in the literature for this task (Bohge et al. 2007, Ergen et al. 2003, Kim et al. 2001, Li et al. 2010, Yin and Liu 2000), where the maximization of the total rate needs to be balanced with the quality-of-service requirements (i.e., rate requirements) of the terminals. Furthermore, the optimization is constrained by the fact that each subcarrier can only be assigned to one terminal during one downlink frame. Hence, bipartite weighted matching has been proposed to compute the allocations at the base station (Kim et al. 2001, Yin and Liu 2000) because it basically maximizes the sum rate of the system but also allows for implicit quality-of-service provisioning. This is possible by adjusting the amount of subcarriers each terminal will receive before the matching is invoked (i.e., copying the vertex representing a specific terminal multiple times into the vertex set U). Based on the average channel quality of the subcarriers toward each terminal, the expected amount of subcarriers required to reach a certain quality-of-service level per allocation phase can be determined (Gross 2009).

However, before utilizing the dynamic allocations for payload transmission, the terminals need to be informed of the subsets of subcarriers allocated to them. Hence, the dynamic allocation of subcarriers to terminals causes an additional signaling overhead that needs to be taken into account. The more allocations are changed from the last downlink frame to the current one, the more overhead has to be spent. Depending on the OFDMA system considered, this reduces the number of symbols that can be used for payload transmission during the upcoming downlink frame. If we denote by $\bar{S}$ the amount of symbols required to signal the overhead, then $S - \bar{S}$ symbols remain for payload transmission. These system characteristics lead to both variants of the matching problems discussed in this paper. If $\bar{S}$ is fixed and therefore only a certain number of assignments can be changed from frame to frame, this leads to the budget-constrained matching problem. On the other hand, if $\bar{S}$ is variable, we end up with elastic reconfiguration costs.

6.2. System Parameters

We evaluate our algorithms regarding two different scenario settings, which we refer to in the following as the *velocity* and *interference* scenarios. Both of these scenario settings are based on a set of common system parameters, which we choose equally and that represent the downlink transmission in 3GPP LTE (Long Term Evolution) OFDMA systems (3rd Generation Partnership Project 2008, Dahlmann et al. 2008). We assume a bandwidth of $B = 20$ MHz, which is subdivided into $n = 96$ subcarriers.[2] Each subcarrier can be assigned individually by the base station. Frame durations are set to $T_f = 1$ (ms), whereas each frame features in total $S = 7$ symbols. For our study, we consider a total of 96 terminals to be present in a single cell. For each of the 96 terminals, the base station has a significant amount of data queued waiting for transmission.

Based on this common set of parameters, we generate channel states for $T = 1{,}000$ consecutive downlink frames for the two different scenario settings in the following way:

- *Velocity scenario*. In the first case, we position the 96 terminals in an area around the base station and consider purely the variation of the object velocity in the propagation scenario. The faster objects move in an propagation environment, the faster the channel states in a wireless system change (Cavers 2000). We vary the object velocity between 1 m/s and 30 m/s. At the same time, the terminals are considered to be relatively far away from the base station, such that their so called signal-to-noise ratio (SNR) on average equals 5 dB. Note that in this case no external interference is present in the system.
- *Interference scenario*. In the second case, we consider terminals to be randomly deployed over a certain area that is served by the base station. However, in contrast to the velocity scenario, in this case there is an interfering base station that causes a degradation of the channel quality. We consider multiple different settings where the interfering base station is closer and closer to the considered set of terminals (varying the distance between the two base stations between 700 and 500 meters).

The generation of the (random) channel states for these two different scenarios follows standard methods commonly applied in wireless systems research. Although the channel instances do not reflect real channel measurements, the considered channel state distributions have been widely used, for instance, in standardization, and can therefore be considered to be realistic. In both scenarios, we focus on a center frequency of 2 GHz. The transmit power per subcarrier is set to 2 W. The noise power per subcarrier is set to -100 dBm. Channel gains are generated based on three effects: path loss, shadowing, and fading (where the first one is only dependent on the distance between transmitter and receiver, whereas the other two effects are random). For path loss we assume a standard model with $10\log(k) = -35.2$ dB and $\alpha = 3.5$. Log-normally distributed shadowing is assumed and parameterized by $\sigma = 4$ dB. Fading is modeled by a Rayleigh-fading random process with a Jakes power spectrum parameterized by a Doppler shift according to the center frequency and the previously described object velocity (which varies in the case of the velocity scenario between 1 m/s and 30 m/s and is fixed to 10 m/s for the interference scenario). Furthermore, we assume an exponential power delay profile with a delay spread of 1 μs.

Based on the channel gains generated according to these assumptions, the resulting SNR γ_{ij}^t per subcarrier/terminal pair is determined in the case of the velocity scenario, whereas for the interference scenario the corresponding signal-to-interference plus noise ratio γ_{ij}^t is determined. These ratios are common metrics to quantify the quality of a specific wireless communication channel[3] and can be directly converted into a corresponding throughput. We assume the Shannon capacity for this relationship, i.e., $w_{ij}^t = \log_2(1 + \gamma_{ij}^t)$. Altogether, this allows us to generate realistic weights w_{ij}^t for the set of T consecutive downlink frames for the two different scenarios.

Based on the weights, the base station now generates the dynamic allocations. Signaling the overhead that stems from the dynamic allocations in LTE consumes, in general, a varying amount of symbols per frame. It is conveyed via the physical downlink control channels Dahlmann et al. (2008). For the setup that we consider, there can be up to $\bar{S} = 3$ symbols used for control information out of the total of $S = 7$ symbols per downlink frame. The control information includes, among other control elements, a terminal identifier, the assigned resource blocks, and the

[2] More precisely, these 96 elements are bundles of subcarriers referred to as resource blocks in LTE. A resource block basically consists of 12 subcarriers each that can be used for payload transmission plus additional subcarriers for channel estimation.

[3] Note that we represent both quantities by the same symbol γ, which is commonly the case in wireless communication research.

modulation/coding scheme used on these resource blocks. There exist different encodings for these different kinds of information. We consider here for illustration purposes a simplified model where per signaling symbol a total of 32 assignment changes can be represented. If the signaling symbols do not indicate a novel assignment of a given subcarrier, the corresponding assignment of the previous phase remains valid.

6.3. Computational Study of the Budget-Constrained Matching

We start presenting and discussing the performance of the introduced algorithms in the context of the budget-constrained matching problem for both scenarios. Because of the budget constraint, we consider that the base station of the system is configured to spend only one symbol on the signaling overhead, i.e., we set $\bar{S} = 1$. This allows the base station to alter at most $k = 32$ allocations from phase to phase. Hence, the objective is to maximize the total net weight

$$\sum_{t=1}^{T} w^t(M^t) \cdot c(M^{t-1}, M^t)$$

with cost function $c(M^{t-1}, M^t)$ defined as

$$c(M^{t-1}, M^t) = \begin{cases} 1 & \text{if } |M^t \cap M^{t-1}| \geq 64, \\ -\infty & \text{otherwise.} \end{cases}$$

This corresponds to the online k-constrained bipartite matching problem where the weights of the bipartite matching graph are revealed in an online fashion (i.e., the base station does not know at time t the weights of the frame at time $t+1, t+2, \ldots$).

6.3.1. Online Algorithms. We consider three different approaches to solve the online k-constrained bipartite matching problem. These different approaches perform per downlink phase the following algorithms:

- *Greedy-MIP*. Per frame the optimal solution to the IP formulation of the k-constrained bipartite matching problem is computed. Recall that the theoretical complexity of this problem is not known yet.
- 1/2-*Approximation*. In this approach, per frame Algorithm 1 is executed. As stated, it represents a viable option to determine the solution to the k-constrained bipartite matching problem. The running time of this algorithm is $O(n^3)$.
- *Lagrange*. Finally, in this approach, per frame a Lagrangian dual problem is solved. This achieves a feasible solution per frame as well but with a significantly higher worst-case running time of $O(n^6)$.

To compare the performance of the above online algorithms, we compute upper bounds using the integer linear programming formulation introduced in §4.2. We relax all binary variables to take on real values to cope with the large programs arising.

6.3.2. Methodology. We consider two different performance metrics for our study. As discussed above, the first one is the net weight of the matchings, which represents the total amount of bits that can be conveyed over the T phases. For illustration and validation purposes, we convert this net weight into the average throughput per terminal obtained by

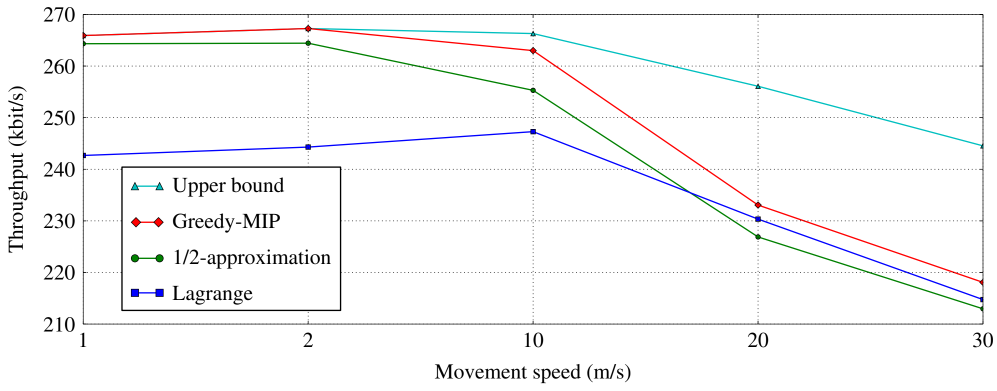
$$(S - \bar{S}) \cdot \frac{\sum_{t=1}^{T} w^t(M^t) \cdot c(M^{t-1}, M^t)}{T \cdot n \cdot T_f}.$$

This is simply a rescaling of the net weight. As second comparison metric, we consider the computation times required to come up with a suitable matching per frame by the different approaches considered.

In the case of the velocity scenario, we vary the maximum object velocity in the propagation scenario. The reason for varying the velocity is that with an increasing velocity, the correlation of the channel states, i.e., edge weights, between two consecutive frames decreases. This is an implicit feature of the fading model introduced in §6.1. To characterize the strength of the correlation, the so-called coherence time of a wireless channel is an established measure in engineering. It quantifies the time span over which the autocorrelation function of the fading process drops below a value of 0.95. In our computational study, we vary the velocity from 1 m/s up to 30 m/s, which corresponds to a decrease of the coherence time from 33 ms down to 1 ms. Recall that a single frame has a time length of 1 ms. Hence, even for large velocities, consecutive edge weights are still correlated, but not as strong as for small velocities. In contrast, in the case of the interference scenario, we vary the position of the interfering base station (between 500 and 700 meters distance from the serving base station). The closer the interfering base station gets, the more terminals are effected by the interring base station. In general, if a wireless channel is interfered with, this leads to a faster change of the channel states compared to the case without interference. Hence, for the interference scenario, the closer the base station gets, the more terminals will experience a worse channel quality in general (due to the interference) while at the same time, for these terminals, also the channel states vary faster, leading to higher signaling costs.

In general, for each setting of the velocities or interference distances, we have evaluated the algorithms' performance over several time lengths, from frame 1 to frame 1,000, for several intervals of 100 frames, and for several intervals of 25 frames. For each algorithm, there is no significant difference in the average net throughput per frame for the different interval lengths, and therefore we present here the results for 10 different runs of 25 frames. We evaluate the algorithms' average performance for frames

Figure 3 (Color online) Average Throughput per Terminal vs. Increasing Channel Variability for the Three Different Approaches for the Budget-Constrained Matching Setting $k = 32$ and the Values of the Upper Bound in the Velocity Scenario



51–75, 151–175, etc. This enables us to compute upper bounds for the corresponding time intervals and to get rid of initial transients in the matching results of the first phases after the initial matching, which is always computed as a maximum weight perfect matching.

6.3.3. Implementation. We used different implementations to compute the results of the different approaches. In case of the 1/2-approximation, the data were processed by a C program that solved the upcoming matchings based on the C-implementation of bipartite weighted matching available in Stachniss (2004). In contrast, the greedy-MIP was obtained via reformulating the problem into a mixed-integer program and solving it with CPLEX (IBM ILOG 2010). The Lagrange approach (Berger et al. 2011) was implemented in C++, and the linear programs arising during the binary search were solved using CPLEX as well. The upper bound on the optimal off-line solution was computed by solving the linear relaxation of the integer program presented in §4.2 using CPLEX.

All schemes were executed on a multicore machine running at 3.3 GHz and having a main memory of 64 GB. The operating system was an Ubuntu 10.10 Linux distribution (64 bit version). Although the machine features several cores, all implementations were single threaded. After executing the corresponding software implementations, the resulting matchings were afterward used for statistical analysis.

6.3.4. Numerical Results: Velocity Scenario. The corresponding results on the average throughput of all four schemes are shown in Figure 3 for the velocity scenario. In addition, Table 1 shows the average values per phase as well as the minimum and maximum terminal throughputs as obtained from all schemes for all runs. In addition, the table also shows the corresponding ratios of the three online schemes compared to the upper bound. Notice initially that the average throughput per terminal decreases as the velocity in the environment increases. This is due to the increasing variation in the channel states from downlink frame to downlink frame, which cannot be fully exploited by all schemes due to the k-constraint. In general, the two heuristic approaches Lagrange and 1/2-approximation achieve almost the same performance as the greedy-MIP approach. For lower speeds, the 1/2-approximation outperforms the Lagrange approach slightly, although for higher speeds this relationship turns around. The two suboptimal schemes are always within 90% of the greedy-MIP solution for the data sets that we have considered. Table 2 shows that the competitive ratios never drop below 80%. This shows, especially for the 1/2-approximation, that for realistic data, the gap to the upper bound is much smaller than its theoretical performance guarantee of $\lfloor k/2 \rfloor / n$ suggests. Note that the upper bound represents a bound on the performance that can be achieved if all assignment decisions

Table 1 Run Times (Milliseconds) of Different Approaches per Phase for the Budget-Constrained Matching According to the System Instances Considered

Movement speed (m/s)	Greedy-MIP			Lagrange			1/2-approximation		
	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
1	168	118	404	431	327	1,139	1.9	1.8	1.9
2	178	119	365	471	343	1,061	1.9	1.9	2.0
10	203	118	648	480	343	1,077	2.2	2.2	2.2
20	201	118	552	515	369	889	2.3	2.3	2.4
30	213	118	803	440	345	728	2.3	2.3	2.3

Table 2 Throughput (Kilobits per Second) of the Different Approaches for the Budget-Constrained Matching According to the System Instances Considered as Well as Competitive Ratios (Second Row for Each Scenario) in the Velocity Scenario

Movement	Greedy-MIP			Lagrange			1/2-approximation			Upper bound		
speed	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
1 (m/s)	266	262	269	243	236	246	263	259	266	266	262	269
%	100	100	100	91.3	89.4	92.1	99.1	98.9	99.3	100	100	100
2 (m/s)	267	265	272	244	241	248	264	262	269	267	265	272
%	100	100	100	91.4	90.9	91.9	98.9	98.7	99.1	100	100	100
10 (m/s)	263	261	265	247	245	250	255	253	258	266	264	269
%	98.8	98.6	98.9	92.9	92.6	93.1	95.9	95.5	96.1	100	100	100
20 (m/s)	233	231	235	230	228	232	227	225	228	256	255	258
%	91.0	90.4	91.5	89.9	89.4	90.7	88.6	88.2	89.1	100	100	100
30 (m/s)	218	215	219	215	212	218	213	211	215	245	244	245
%	89.2	88.3	89.7	87.8	86.8	88.9	87.1	86.5	87.7	100	100	100

are optimally computed with complete knowledge of the future states of the subcarrier/terminal pairs. Moreover, the upper bound is the optimal value of the linear programming relaxation; therefore, the optimal integral solution may even have a lower value for the overall throughput. Interestingly, having (and using) this knowledge can thus at most achieve a performance improvement of up to 20%. We conclude that for the communication system considered (as well as its parameterization), the established signaling system is already quite efficient, as the usage of statistical information regarding the future channel evolution would only yield a marginal additional performance improvement.

For our implementations, we have also traced the computation times, which are summarized as averages (as well as minimum and maximum values) in Table 2. The values show clearly that the 1/2-approximation is indeed a good trade-off between the achieved performance and the running times. As mentioned previously, run times in the range of milliseconds qualify an algorithm to be applied in a real OFDMA system, as the algorithms can be further tuned to have a run time significantly below 1 ms by implementing subroutines of the algorithms in hardware and/or optimizing the software implementation itself. Also note that especially the worst case running time of the 1/2-approximation clearly outperforms the other approaches. Because of the large computational overhead of the Lagrange approach, we do not consider it further for the elastic reconfiguration costs.

6.3.5. Numerical Results: Interference Scenario. In Figure 4, we present the average throughput results for the four different schemes in the case of the interference scenario, whereas in Table 3 the corresponding values are shown in addition to the minimum and maximum values as well as well the competitive ratios. Notice that in the plot we also show the results of the system performance in the case where no interferer is present (right set of bar plots). In general, we observe that as the interference in the cell becomes stronger, the performance of all schemes degrades. However, whereas the 1/2-approximation is in all considered cases very close to the greedy-MIP performance and within 85% of the performance of the upper bound, the performance of the Lagrange approach decreases more strongly as the interference in the cell increases and achieves only around 70% of the performance of the upper bound.

6.4. Computational Study for Elastic Reconfiguration Costs

The case of elastic reconfiguration costs results from considering a system setup where from downlink frame to downlink frame a varying number of symbols can be consumed by the signaling overhead. According to our model for the signaling channel, a minimum of $\bar{S} = 0$ and a maximum of $\bar{S} = 3$ symbols will be considered in the following. The objective function of interest is again

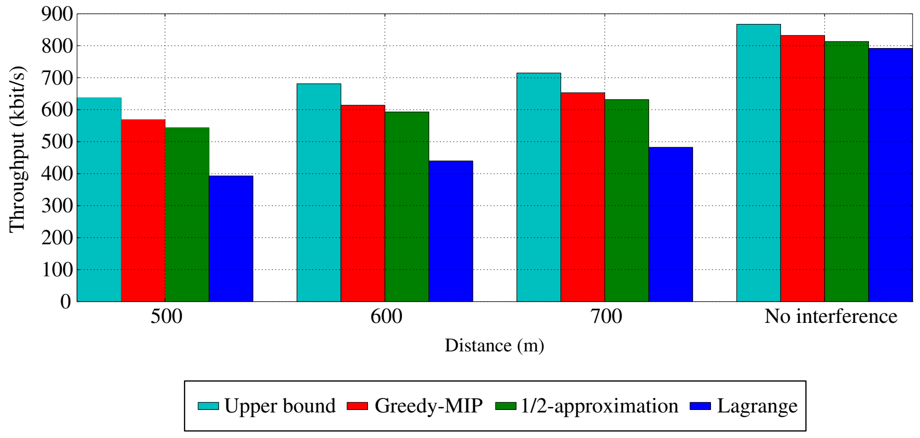
$$\sum_{t=1}^{T} w^t(M^t) \cdot c(M^{t-1}, M^t).$$

However, the cost function $c(M^{t-1}, M^t)$ is given according to the elastic reconfiguration cost model by

$$c(M^{t-1}, M^t) = \frac{4}{7} + \frac{1}{7} \cdot \left\lfloor \frac{|M^t \cap M^{t-1}|}{32} \right\rfloor.$$

Note that this reconfiguration cost model is slightly different than the one discussed in §5, where the reconfiguration costs were linearly dependent on the amount of modified assignments from phase to phase. However, for the considered system, the reconfiguration costs depend on a discretized amount of modified assignment changes from phase to phase, which results in the above cost model. Although from a formal point of view this is a slightly different cost model, we nevertheless apply our algorithms and

Figure 4 (Color online) Average Throughput per Terminal vs. Increasing Distance of the Interfering Base Station for the Three Different Approaches for the Budget-Constrained Matching Setting $k = 32$ and the Values of the Upper Bound in the Interference Scenario



results from §5 because we expect only very little modification from an exact analysis. Note that in the above definition for the reconfiguration costs, $|M^t \cap M^{t-1}| \le 96$, and thus $0 \le c(M^{t-1}, M^t) \le 1$.

6.4.1. Online Algorithms. For elastic reconfiguration costs we consider the following three different approaches:

• *Maximum weight matching*. Here, per frame, the solution to the unrestricted weighted matching instance is computed. With elastic reconfiguration costs, this always leads to a feasible matching; however, it does not consider the number of changes in consecutive assignments and is therefore not sensible to the net weight. Note that this approach has again a running time of $O(n^3)$.

• *Greedy-MIP*. Per frame, the optimal solution to the IP formulation of the k-constrained bipartite matching problem is computed sequentially setting $k = 32$, then $k = 64$, and finally $k = 96$. Out of the different versions, the best one is then chosen for comparison purposes.

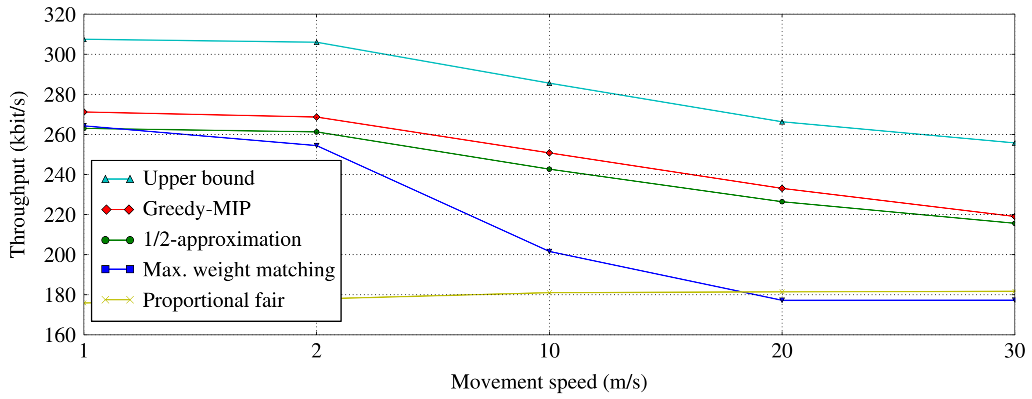
• 1/2-*Approximation*. In this approach, per frame, Algorithm 1 is executed sequentially in the same manner as with the greedy-MIP approach. In principle, this approach has a complexity of $O(n^4)$ if the reconfiguration costs scaled linearly. However, for the considered system, the reconfiguration costs can only take three values such that the complexity remains at $O(n^3)$.

• *Proportional fair scheduling* (*PFS*). In addition to the above three approaches, we also run simulations using a standard resource assignment algorithm from literature. This is the well-known proportional fair scheduler (Kelly et al. 1998, Kim and Han 2005), which represents an algorithmic compromise between opportunistically assigning resources to the terminal with the best channel states (leading to the highest sum throughput, but to an unfair allocation of the rates) and max-min fairness (where all terminals are required to be assigned the exact same rate providing perfect fairness but achieving possibly a low sum throughput). PFS does not necessarily compute a perfect matching in each time slot. Instead, per terminal j, the average throughput r_j^θ over the last θ time slots is computed, and the current channel states $w_{i,j}$ are normalized by the average r_j^θ. Once this modification of the channel coefficients is done, each subcarrier is assigned to the terminal with the highest normalized channel coefficient, i.e., in a greedy

Table 3 Throughput (Kilobits per Second) of the Different Approaches for the Budget-Constrained Matching According to the System Instances Considered as Well as Competitive Ratios (Second Row for Each Scenario) in the Interference Scenario

Interferer (int.)	Greedy-MIP			Lagrange			1/2-approximation			Upper bound		
distance	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
500 (m)	569	564	577	393	377	409	544	540	548	637	631	644
%	89	89	90	62	60	64	85	86	85	100	100	100
600 (m)	614	609	621	440	430	452	593	587	601	681	676	688
%	90	90	90	65	64	66	87	87	87	100	100	100
700 (m)	653	646	660	482	490	519	632	628	638	715	711	721
%	91	91	92	67	66	69	88	88	88	100	100	100
No int.	832	830	836	792	688	834	813	809	817	867	864	871
%	96	96	96	91	80	96	94	94	94	100	100	100

Figure 5 (Color online) Average Throughput per Terminal vs. Increasing Channel Variability for the Four Different Approaches for the Elastic Reconfiguration Costs and the Values of the Upper Bound in the Velocity Scenario



fashion. By exploiting this kind of "memory," terminals with low average throughput are weighted automatically by higher channel coefficients, even if the absolute channel states are fairly poor. As a result, terminals receive average rates over time that are in proportion to their own average channel states, but also in proportion to the average channel states of all other terminals. Note that proportional fair scheduling does not take the signaling costs into account. In our evaluation, we set the window θ over which the rates are averaged to 100 slots.

To compare the performance of the above online algorithms, we again compute upper bounds using the integer linear programming formulation introduced in §5.2. To cope with the very large integer programs that arise, the results of the upper bound presented here are obtained by solving the following linear relaxation of the formulation, where (15) and (16) are replaced by (17) and (18):

$$\sum_{i,j} z_{ij}^t \leq n \cdot \sum_{i,j} y_{ij}^t \quad \text{for all } t \geq 2, \tag{17}$$

$$z_{ij}^t, y_{ij}^t, x_{ij}^t \geq 0 \quad \text{for all } i, j \in [n],\ t \geq 1. \tag{18}$$

6.4.2. Methodology. Most of the methodology is taken from the study on budget-constrained matchings. Again we scale the net weight obtained from all matchings by a factor given by

$$S \cdot \frac{\sum_{t=1}^{T} w^t(M^t) \cdot c(M^{t-1}, M^t)}{T \cdot n \cdot T_f}$$

to obtain the average throughput per terminal. As a second performance metric we consider again the run times of the algorithms. However, in contrast to the budget-constrained matching case, we consider two more metrics. On the one hand, for flows of packets, the so-called quality of service is a further important metric. This can be quantified, for example, by the minimum throughput that a terminal receives over time. It represents the fact that in the case of delay-sensitive applications (like voice or video), a minimum throughput must be provided to each terminal. In the following, we consider here the minimum throughput over 10 consecutive slots. As a further metric we consider the variance over the assigned average throughput per terminal (Jain 1991). This metric shows how evenly (and hence "fair") the throughput is distributed among the terminals for the different algorithms.

As in the budget-constrained matching case, we consider again two scenarios. In the case of the velocity scenario, the parameter varied is again the maximum object velocity of the propagation environment with the same speed settings as in §6.3.2. For the interference scenario, we again set the interfering base station closer and closer to the serving base station with the same distance as mentioned above. Implementation-wise, the different approaches rely on the programs discussed in the above section and are executed on the same computer as mentioned above.

6.4.3. Numerical Results: Velocity Scenario. In Figure 5 we show the average throughput of the five different schemes for the velocity scenario. In addition, these values are also shown in Table 4 together with the minimum and maximum values obtained over the different runs (we also show the ratios of the four online schemes compared to the upper bound). Note that the graphs again have a decreasing slope as the velocity of the terminals increases. Furthermore, whereas the 1/2-approximation is quite close to the greedy-MIP approach (within 3%), the maximum weight matching algorithm suffers significantly from the additional signaling burden, which leads to a performance loss of up to 25% compared to the greedy-MIP approach. The figure also reveals the potential performance gain that might be obtained from predicting future channel states and choosing assignments for example based on such predictions

Table 4 **Throughput (Kilobits per Second)of the Different Approaches for the Elastic Reconfiguration Costs According to the System Instances Considered in the Velocity Scenario**

Movement	Greedy-MIP			Max weight M			PFS			1/2-approximation			Upper bound		
speed	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
1 (m/s)	271	265	279	264.3	259.8	266.9	176.0	172.7	176.0	263.0	259.7	267.5	307.5	303.2	310.8
%	88.2	86.9	90.8	86.0	85.3	86.5	57.2	57.0	56.6	85.5	84.0	87.2	100	100	100
2 (m/s)	269	264	273	254.4	248.7	259.1	177.9	175.5	177.9	261.3	256.3	266.2	306.0	303.1	311.8
%	87.8	87.0	89.2	83.1	81.9	84.9	58.1	57.9	57.1	85.4	84.2	86.2	100	100	100
10 (m/s)	251	246	253	201.6	199.0	204.8	181.1	179.5	181.1	242.7	239.7	245.8	285.6	282.7	288.1
%	87.8	87.1	88.5	70.6	69.1	72.2	63.4	63.5	62.9	85.0	84.5	85.3	100	100	100
20 (m/s)	233	231	236	177.2	176.3	178.4	181.5	180.5	181.5	226.4	224.5	228.8	266.3	256.2	269.5
%	87.5	86.7	90.9	66.6	66.2	69.1	68.2	70.5	67.3	85.0	84.0	88.7	100	100	100
30 (m/s)	219	217	221	177.3	176.8	177.8	181.8	181.0	181.8	215.7	213.6	218.1	255.8	255.1	256.5
%	85.6	85.1	86.0	69.3	69.1	69.6	71.1	71.0	70.9	84.3	83.4	85.2	100	100	100

Table 5 **Quality-of-Service (QoS) Support (Kilobits per Second) and Fairness (Kilobits per Second Squared) of the 1/2-Approximation Compared to the Proportional Fair Scheduler for the Elastic Configuration Costs Over Different Object Speeds in the Propagation Scenario in the Velocity Scenario**

Movement	1/2-approximation		PFS	
speed (m/s)	QoS (min rate)	Fairness (σ^2)	QoS (min rate)	Fairness (σ^2)
1	560	71	0	17
2	979	36	0	7.6
10	1,645	11	0	2
20	1,570	9	0	1
30	1,411	6	0	1

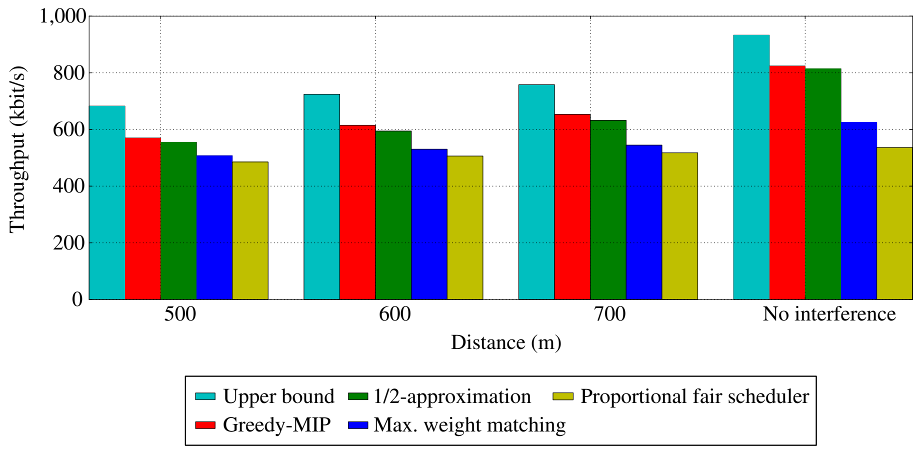
or other statistical knowledge (anticipating the signaling costs). Such schemes are limited by the performance of the upper bound, which achieves about 20% more average throughput than the greedy-MIP strategy. This shows nicely that a greedy approach in general is not far away from an off-line optimum that can be achieved if the channel states of all T frames are already known. Finally, note that the proportional fair scheduling approach, which we use as a reference scheme here, has the worst performance of all schemes, featuring a constant average throughput per terminal over all terminal speeds. We refine this initial analysis by considering two further performance metrics quantifying fairness and quality of service (in the form of the minimum assigned rate over all terminals) in Table 5. This analysis further reveals the trade-off encountered in OFDMA resource scheduling. Because of the perfect matching characteristic of the approximation, per slot, every terminal receives a subcarrier, and hence the scheme achieves a good fairness; i.e., the minimum rate over 10 slots is still reasonably high (but by about a factor of four lower than the average rate over 10 slots). In contrast, the proportional fair scheduling algorithm has a relatively low quality-of-service support, as for all scenarios periods of 10 slots can be found for which the minimum rate turns out to be 0. This is partially due to the fact that there are quite many terminals compared to the number of resource blocks. On the other hand, if considering fairness, we observe from Table 5 that (as expected) proportional fair scheduling achieves a more even distribution of the long-term average throughput per terminal, compared to the approximation. Note that this comes at the price of a lower aggregate throughput, as shown in Figure 5.

Regarding the run times, we again observe a much lower average value in case of the 1/2-approximation compared to the greedy-MIP in Table 6. The lower run time of the maximum weight matching approach results from the fact that, per instance, only a single matching of cardinality 96 needs to be computed. Although this run time is quite low, the

Table 6 **Run Times (Milliseconds) of the Different Approaches per Phase for the Elastic Reconfiguration Costs According to the System Instances Considered**

Movement	Greedy-MIP			Max weight M			PFS			1/2-approximation		
speed (m/s)	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
1	548	402	1,445	0.4	0.4	0.5	0.1	0.1	0.1	4.6	4.5	4.8
2	534	408	1,086	0.4	0.4	0.5	0.1	0.1	0.1	4.7	4.6	4.8
10	544	410	1,048	0.4	0.4	0.5	0.1	0.1	0.1	4.8	4.8	4.9
20	548	412	1,204	0.4	0.4	0.4	0.1	0.1	0.1	4.9	4.9	5.0
30	589	416	2,066	0.4	0.4	0.4	0.1	0.1	0.1	4.9	4.8	5.0

Figure 6 (Color online) Average Throughput per Terminal vs. Increasing Distance of the Interfering Base Station for the Four Different Approaches for the Elastic Reconfiguration Costs, and the Values of the Upper Bound in the Interference Scenario



performance is worse than the one of the 1/2-approximation. Because of its simplicity, the proportional fair scheduler has the lowest run time of all considered schemes; nevertheless, it also provides the lowest performance in terms of average throughput as well as quality-of-service provisioning.

6.4.4. Numerical Results: Interference Scenario. In Figure 6, we show the average throughput of the four different schemes plus the upper bound in the case of the interference scenario for elastic reconfiguration costs, whereas Table 7 shows the corresponding numerical values including the minimum, the maximum, and the competitive ratios. As the interfering base station gets closer and closer to the serving base station, the increase in interference leads to a decreasing average throughput for all considered schemes. As with the budget-constrained case, we observe again that the 1/2-approximation achieves in all cases a performance very close to that of the greedy-MIP scheme. However, we notice a slightly bigger gap to the upper bound (compared to the budget-constrained case for the interference scenario). However, we also observe this increased gap for elastic reconfiguration costs in the case of the velocity scenario. Furthermore, the stronger the interference gets in the cell, the lower the performance advantage of the 1/2-approximation and of the greedy-MIP approaches to maximum weight matching and the proportional fair scheduler. This is clearly due to the increased channel variability as the interference increases, which causes larger costs in terms of the signaling overhead for the greedy-MIP and the 1/2-approximation in comparison to the overhead-insensitive allocation schemes (maximum weight matching and proportional fair scheduling).

Finally, in Table 8 we give the results on the fairness and quality-of-service support of the 1/2-approximation and the proportional fair scheduler for the investigated interference scenario. These results reaffirm the findings of the velocity scenario. Whereas the 1/2-approximation has a better quality-of-service support for all considered scenario settings, the fairness is in general better for the proportional fair scheduler. Note that the fairness measure—which is the variance over the average throughput per terminal in our case—is in general much higher in the interference scenarios than in the velocity scenario due to the positioning of the terminals (which are spread all over the cell in the case of the interference scenario, leading

Table 7 Throughput (Kilobits per Second) of the Different Approaches for the Elastic Reconfiguration Costs According to the System Instances Considered in the Interference Scenario

Interferer (int.)	Greedy-MIP			Max weight *M*			PFS			1/2-approximation			Upper bound		
dist.	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
500 (m)	570	565	575	508	485	528	485	479	494	555	549	563	682	676	690
%	88	88	89	74	72	77	71	71	71	81	81	82	100	100	100
600 (m)	615	609	620	530	509	550	506	500	512	595	590	599	724	718	731
%	85	85	85	73	71	75	70	70	70	82	82	82	100	100	100
700 (m)	653	646	660	545	533	553	518	510	526	632	626	637	758	752	765
%	86	86	86	72	71	72	68	68	69	83	83	83	100	100	100
No int.	824	817	829	625	603	652	536	462	596	814	804	823	933	930	936
%	88	88	89	67	65	70	57	50	64	87	86	88	100	100	100

Table 8 Quality-of-Service (QoS) Support (Kilobits per Second) and Fairness (Kilobits per Second Squared) of the 1/2-Approximation Compared to the Proportional Fair Scheduler for the Elastic Configuration Costs for Different Interferer Distances in the Interference Scenario

	1/2-approximation		PFS	
Interferer (int.) dist.	QoS (min rate)	Fairness (σ^2)	QoS (min rate)	Fairness (σ^2)
500 (m)	1,673	36,295	0	17,136
600 (m)	2,260	41,531	0	16,141
700 (m)	2,784	40,976	0	15,401
No int.	2,222	42,969	0	13,307

to a much higher variance of the average throughput). Because the average throughput per terminal is higher in case of the 1/2-approximation, we conclude that the 1/2-approximation is also a good approach for interference scenarios.

7. Conclusions

In this paper we addressed resource assignment problems with additional constraints motivated by applications in which assignments have to be realized over time and the reconfiguration of any two consecutive assignments has a significant impact on the overall performance. For our target applications, the run time of matching algorithms is extremely important, having an acceptable range of a few milliseconds only. We introduced two variants of bipartite matching problems with reconfiguration costs and provided extremely fast approximation and online algorithms with provable approximation guarantees and competitive ratios. We also tested our algorithms on realistic instances in the context of the downlink of an OFDMA wireless cell. It turned out that for the instances considered, our algorithms perform extremely well with respect to both solution quality and running time: the solution quality, on average, is within a factor of 0.8–0.9 of optimal off-line solutions, and the running times are at most 5 ms per phase even in the worst case. Thus, our algorithms are well suited to be applied in the context of OFDMA systems.

Supplemental Material

Supplemental material to this paper is available at http://dx.doi.org/10.1287/mnsc.2015.2221.

Acknowledgments

An extended abstract containing parts of §§3 and 4 appeared in the *Proceedings of the 2010 IEEE INFOCOM Conference* (Berger et al. 2010). This research was funded in part by the Deutsche Forschungsgemeinschaft (DFG) Cluster of Excellence on Ultra High-Speed Information and Communication, German Research Foundation [Grant DFG EXC 89]. The research of T. Harks was supported by the Marie-Curie grant "Protocol Design" [No. 327546], funded within FP7-PEOPLE-2012-IEF.

References

3rd Generation Partnership Project (2008) Evolved Universal Terrestrial Radio Access (EUTRA); Radio Frequency (RF) system scenarios. Technical Report 36.942, 3rd Generation Partnership Project, ETSI.

Berger A, Gross J, Harks T (2010) The *k*-constrained bipartite matching problem: Approximation algorithms and applications to wireless networks. *Proc. 29th IEEE Internat. Conf. Comput. Comm.* (IEEE Computer Society, Washington, DC), 2043–2051.

Berger A, Bonifaci V, Grandoni F, Schäfer G (2011) Budgeted matching and budgeted matroid intersection via the gasoline puzzle. *Math. Programming* 128(1):355–372.

Bohge M, Gross J, Meyer M, Wolisz A (2007) Dynamic resource allocation in OFDM systems: An overview of cross-layer optimization principles and techniques. *IEEE Network* 21(1):53–59.

Borodin A, El-Yaniv R (1998) *Online Computation and Competitive Analysis* (Cambridge University Press, Cambridge, UK).

Buchbinder N, Lewin-Eytan L, Menache I, Naor J, Orda A (2012) Dynamic power allocation under arbitrary varying channels—An online approach. *IEEE/ACM Trans. Networking* 20(2): 477–487.

Cavers JK (2000) *Mobile Channel Characteristics* (Kluwer Academic Publishers, Norwell, MA).

Dahlmann E, Parkvall S, Sköld J, Beming P (2008) *3G Evolution: HSPA and LTE for Mobile Broadband* (Academic Press, Oxford, UK).

Ergen M, Coleri S, Varaiya P (2003) QoS aware adaptive resource allocation techniques for fair scheduling in OFDMA based broadband wireless access systems. *IEEE Trans. Broadcasting* 49(4):362–370.

Fiat A, Woeginger GJ, eds. (1998) *Online Algorithms, The State of the Art*, Lecture Notes in Computer Science, Vol. 1442 (Springer–Verlag, Berlin).

Fu A, Modiano E, Tsitsiklis J (2006) Optimal transmission scheduling over a fading channel with energy and deadline constraints. *IEEE Trans. Wireless Comm.* 5(3):630–641.

Galil Z (1986) Efficient algorithms for finding maximum matching in graphs. *ACM Comput. Surveys* 18(1):23–38.

Goudreau M, Kolliopoulos S, Rao S (2000) Scheduling algorithms for input-queued switches: Randomized techniques and experimental evaluation. *Proc. IEEE Internat. Conf. Comput. Comm.* (IEEE Computer Society, Washington, DC), 1634–1643.

Gross J (2009) Admission control based on OFDMA channel transformations. *Proc. 10th IEEE Internat. Sympos. World of Wireless, Mobile and Multimedia Networks* (IEEE Computer Society, Washington, DC), 1–11.

Gross J, Geerdes H, Karl H, Wolisz A (2006) Performance analysis of dynamic OFDMA systems with inband signaling. *IEEE J. Selected Areas Comm.* 24(3):427–436.

Gusfield D, Irving RW (1989) *The Stable Marriage Problem: Structure and Algorithms* (MIT Press, Cambridge, MA).

Henttonen T, Aschan K, Puttonen J, Kolehmainen N, Kela P, Moisio M, Ojala J (2008) Performance of VoIP with mobility in UTRA long term evolution. *Proc. IEEE Vehicular Tech. Conf.* (IEEE Computer Society, Washington, DC), 2492–2496.

Höhn W, König FG, Möhring RH, Lübbecke ME (2011) Integrated sequencing and scheduling in coil coating. *Management Sci.* 57(4):647–666.

IBM ILOG (2010) IBM ILOG CPLEX 12.0 User's Manual. IBM ILOG, Paris, France.

Jain R (1991) *The Art of Computer Systems Performance Analysis* (John Wiley & Sons, New York).

Kelly F, Maulloo A, Tan D (1998) Rate control for communication networks: Shadow price proportional fairness and stability. *J. Oper. Res. Soc.* 49(3):237–252.

Kim H, Han Y (2005) A proportional fair scheduling for multicarrier transmission systems. *IEEE Comm. Lett.* 9(3):210–212.

Kim WY, Kak AC (1991) 3-D object recognition using bipartite matching embedded in discrete relaxation. *IEEE Trans. Pattern Anal. Machine Intelligence* 13(3):224–251.

Kim I, Lee H, Kim B, Lee Y (2001) On the use of linear programming for dynamic subchannel and bit allocation in multiuser OFDM. *Proc. IEEE Global Telecomm. Conf.* (IEEE Computer Society, Washington, DC), 3648–3652.

Knuth DE (1976) *Mariages Stables* (Les Presses de l'Universite de Montreal, Montreal).

Korte B, Vygen J (2000) *Combinatorial Optimization* (Springer–Verlag, Berlin).

Kuhn H (1955) The Hungarian method for the assignment problem. *Naval Res. Logist. Quart.* 2(1–2):83–97.

Li W, Zhang Y, So A, Win M (2010) Slow adaptive OFDMA systems through chance constrained programming. *IEEE Trans. Signaling Processing* 58(7):3858–3869.

Midran R, Boyd S, Lall S (2010) Fast algorithms for resource allocation in wireless cellular networks. *IEEE/ACM Trans. Networking* 18(3):973–984.

Papadimitriou C, Yannakakis M (1982) The complexity of restricted spanning tree problems. *J. ACM* 29(2):285–309.

Papadimitriou C, Yannakakis M (2000) On the approximability of trade-offs and optimal access of web sources. *Proc. Annual Sympos. Foundations Comput. Sci.* (IEEE Computer Society, Washington, DC), 86–92.

Schrijver A (2003) *Combinatorial Optimization. Polyhedra and Efficiency* (Springer–Verlag, Berlin).

Stachniss C (2004) C implementation of the Hungarian method. Last accessed October 27, 2015, http://www.informatik.uni-freiburg.de/~stachnis/misc.html.

Urgaonkar R, Neely M (2009) Opportunistic scheduling with reliability guarantees in cognitive radio networks. *IEEE Trans. Mobile Comput.* 8(6):766–777.

Yin H, Liu H (2000) An efficient multiuser loading algorithm for OFDM-based broadband wireless systems. *Proc. IEEE Global Telecomm. Conf.* (IEEE Computer Society, Washington, DC), 103–107.

Zhang Z, Yang Y (2004) Optimal scheduling algorithms in WDM optical interconnects with limited range wavelength conversion capability. *IEEE Trans. Parallel and Distributed Systems* 15(11):1012–1026.

Zhao C, Zou M, Shen B, Kim B, Kwak K (2008) Cooperative spectrum allocation in centralized cognitive networks using bipartite matching. *Proc. IEEE Global Telecomm. Conf.* (IEEE Computer Society, Washington, DC), 1–6.