MISTA 2013

A Decomposition of the Max-min Fair Curriculum-based Course Timetabling Problem: The Impact of Solving Subproblems to Optimality

Moritz Mühlenthaler · **Rolf Wanka**

Abstract We propose a decomposition of the max-min fair curriculum-based course timetabling (MMF-CB-CTT) problem. The decomposition models the room assignment subproblem as a generalized lexicographic bottleneck optimization problem (GLBOP). We show that the GLBOP can be solved in polynomial time if the corresponding sum optimization problem can be solved in polynomial time as well. Thus, the room assignment subproblem of the MMF-CB-CTT problem can be solved efficiently. We apply this result to a previously proposed heuristic algorithm for the MMF-CB-CTT problem, in which solving the room assignment subproblem is a key ingredient. Our experimental results indicate that using the proposed decomposition improves the performance of the algorithm on most of the 21 ITC2007 test instances with respect to the quality of the best solution found and the average solution quality. Furthermore, we introduce a measure for the quality of a solution to a (generalized) lexicographic bottleneck optimization problem. This measure helps to overcome some limitations imposed by the qualitative nature of max-min fairness and aids the statistical evaluation of the performance of randomized algorithms for such problems.

1 Introduction

Many combinatorial problems can be solved effectively by decomposing them into subproblems which are tackled individually. Natural examples of successful decomposition approaches are divide-and-conquer algorithms for sorting and searching [19]. For many hard resource allocation and scheduling problems there is also some gain in dividing them into subproblems, even if the subproblems themselves are hard to solve or interact in some way. Basically, any approach that solves such a problem in two or more steps implicitly or explicitly decomposes it into subproblems. In the literature, there are numerous examples of multi-step and other decomposition approaches to solving such combinatorial problems; applications include timetabling [7,21], crew and staff scheduling [35,11], and production scheduling [14,16].

Research funded in parts by the School of Engineering of the University of Erlangen-Nuremberg.

Moritz Mühlenthaler · Rolf Wanka
University of Erlangen-Nuremberg, Department of Computer Science
E-mail: {moritz.muehlenthaler | rolf.wanka}@cs.fau.de

We consider a decomposition approach to a variant of the curriculum-based course timetabling (CB-CTT) problem. The CB-CTT problem has been proposed in the context of the timetabling competition ITC2007 [25] and has since then received a great deal of attention in the timetabling community. The CB-CTT problem models the task of assigning timeslots and rooms to courses in the setting of a university and includes a number of requirements that typically arise in real-world course timetabling applications. For instance, courses occurring in the same curriculum must not be taught at the same time and courses should be assigned to rooms with a sufficient number of seats. The combinatorial structure of the problem is quite complex and has been investigated, for example, using polyhedral studies [8,21]. Due to the structural complexity, solving instances with a large number of courses and curricula is typically out of reach for exact methods. However, a wealth of (meta-)heuristic methods have been applied successfully to generate high quality timetables, even for large CB-CTT instances, see for example [1,23,27]. In this work, our focus is the max-min fair curriculum-based course timetabling (MMF-CB-CTT) problem introduced in [26], which replaces the original optimization objective by a lexicographic bottleneck objective. The goal of the MMF-CB-CTT problem formulation is to favor *fair* timetables and thus improve the overall stakeholder satisfaction. The underlying fairness concept is (lexicographic) max-min fairness.

It is a common technique to decompose the CB-CTT problem into subproblems which are easier to handle individually [21,23]. The usual approach is to perform room and timeslot assignments separately, but other approaches have been explored as well [7]. Applying the usual decomposition approach yields as subproblems a bounded list coloring problem for the timeslot assignment and, for each timeslot, a linear sum assignment problem (LSAP) for assigning courses to rooms [8,21]. Unfortunately, there are dependencies between LSAPs for different timeslots, so an optimal room assignment can only be obtained for a single timeslot, while the rest of the timetable remains fixed. We show that for an analogous decomposition of the MMF-CB-CTT problem, the room assignment subproblem for a single timeslot can also be solved efficiently by modeling it as a generalized lexicographic bottleneck assignment problem. In a more general setting, we show that generalized lexicographic bottleneck optimization problems can be solved efficiently if the corresponding sum optimization problem can be solved efficiently.

Furthermore, we propose a new measure for the quality of a solution to an optimization problem with a max-min fairness/lexicographic bottleneck objective such as the MMF-CB-CTT problem. This measure helps to overcome some limitations imposed by the qualitative nature of max-min fairness.

We evaluate the quality of the timetables produced by the algorithm MAXMINFAIR_SA from [26], with and without the new decomposition. We use our aforementioned new quality measure to compute the average solution quality. In the original algorithm the room assignment subproblem was modeled as an LSAP. However, an optimal solution to the LSAP is not necessarily optimal for the room assignment subproblem of the MMF-CB-CTT problem. Our experiments indicate that making use of the new decomposition improves the best timetables found on 18 out of 21 CB-CTT instances from the ITC2007 competition. The new decomposition results in an improved average solution quality for 16 out of 21 instances. According to the Wilcoxon rank-sum test (one-sided, significance level 0.01) MAXMINFAIR_SA with the new decomposition is significantly better than the original approach on 12 instances.

The remainder of this work is organized as follows: In Section 2, we provide relevant background on the CB-CTT and MMF-CB-CTT problem formulations, as well as max-min fairness and the assignment problem. In Section 3, we introduce the decomposition of the

MMF-CB-CTT problem. We propose the measure for the amount of max-min fairness of an allocation in Section 4. Section 5 presents our evaluation of the performance impact of the decomposition for MMF-CB-CTT problems. Section 6 contains concluding remarks.

2 Background

In this section we will provide some relevant background on fair resource allocation, university course timetabling and the assignment problem.

2.1 Fair Resource Allocation

Fairness comes into play when scarce resources are distributed over a set of stakeholders with demands. The topic of fairness and how to measure it has received great attention for example in the area of economics, where the distribution of wealth and income is of interest [12]. In computer science, fairness aspects have been studied for example in the design of network communication protocols, in particular in the context of bandwidth allocation and traffic shaping [22, 15]. Fairness aspects have been addressed explicitly for example for various kinds of scheduling problems, including personnel scheduling [33], sports scheduling [31], course scheduling [26] and aircraft scheduling [34]. In the context of resource allocation allocation in general, fairness has been studied in [3, 28]. Approximation algorithms for fair optimization problems have been studied in [18, 20].

In the next sections of this work we will deal with a fair variant of a university course timetabling problem from [26] that builds on the notion of max-min fairness. A resource allocation is called *max-min fair* if an improvement for any stakeholder is possible only at the expense of another stakeholder who is worse-off. Consider the problem of allocating resources to n stakeholders. A resource allocation induces an allocation vector $X = (x_1, \ldots, x_n)$, where x_i, $1 \leq i \leq n$, corresponds to the amount of resources allocated to stakeholder i. Let $\vec{X}$ denote the sequence containing the entries of an allocation vector X sorted in non-decreasing order. Likewise, let $\overleftarrow{X}$ denote the sequence containing the entries of X sorted in non-increasing order. For allocation vectors X and Y induced by two solutions to a minimization (maximization) problem, we write $X \preceq Y$ ($X \preceq_{max} Y$) if X is at least as fair as Y with respect to max-min fairness. That is, $X \preceq Y$ holds iff $\vec{X} \preceq_{lex} \vec{Y}$, and $X \preceq_{max} Y$ holds iff $\vec{Y} \preceq_{lex} \vec{X}$. A solution to a resource allocation problem is called *max-min fair* if its induced allocation vector is at least as fair as the induced allocation vector of any other feasible solution.

Max-min fairness enforces an efficient resource usage to some extent, since an improved resource utilization is accepted to the benefit of a stakeholder as long as it is not at the expense of another stakeholder who is worse-off. Hence, a max-min fair solution to a resource allocation problem is Pareto-optimal. One of the limitations of max-min fairness is that the concept is purely qualitative, i. e., given two allocation vectors X and Y, max-min fairness just determines which of the two is fairer, but not by how much. In order to aid the statistical evaluation of the performance of algorithms for max-min fair resource allocation problems, in Section 4 we will introduce a metric for the difference in quality between two allocation vectors which is compatible with the $\preceq$-relation.

2.2 Curriculum-based Course Timetabling

The academic course timetabling problem captures the task of assigning a set of courses to rooms and timeslots in the setting of a university. In Section 3 we will focus on decompositions of two particular variants of the academic course timetabling problem: the curriculum-based course timetabling (CB-CTT) problem from track three of the second international timetabling competition [10], and its max-min fair version, MMF-CB-CTT, proposed in [26]. The CB-CTT formulation has attracted a great deal of interest in the research community as well as in practical course timetabling [23,21,32]. The max-min fair variant differs from the basic CB-CTT formulation only with respect to the objective function. We will now introduce some terminology and state definitions relevant to the later sections of this work.

A CB-CTT instance consists of a set of courses, a set of curricula, a set of rooms, a set of teachers and a set of days. Each day is divided into a fixed number of timeslots; a day together with a timeslot is referred to as a *period*. A period in conjunction with a room is called a *resource*. Each course consists of a set of events that need to be scheduled, is taught by a teacher and has a fixed number of students attending it. A course can only be taught in certain available periods. Each curriculum is a set of courses, no two of which may be taught in the same period. Each room has a *capacity*, a maximum number of students it can accommodate. A solution to a CB-CTT instance is a *timetable*, i. e., an assignment of the courses to the resources subject to a number of hard and soft constraints. A timetable that satisfies all hard constraints is *feasible*.

Later on, we will deal exclusively with feasible timetables, so we will not cover the evaluation of hard constraints at all (please refer to [10] a detailed description). However, some understanding of the soft constraint evaluation will be useful later on, so we will touch on this very briefly. The CB-CTT problem formulation features the following soft constraints:

S1 *RoomCapacity*: Each lecture should be assigned to a room of sufficient size.
S2 *MinWorkingDays*: The individual lectures of each course should be distributed over a certain minimum number of days.
S3 *IsolatedLectures*: For each curriculum, all courses in the curriculum should be scheduled in adjacent periods.
S4 *RoomStability*: The lectures of each course should be held in the same room.

The violation of a soft constraint results in a "penalty" for the timetable. The total penalty of a timetable τ is aggregated by the objective function c which just sums the penalties for the individual soft constraint violations:

$$c(\tau) = \sum_{1 \leq i \leq 4} c_{Si}(\tau), \tag{1}$$

where $c_{S1}, \ldots, c_{S4}$ are the penalties determined by the soft constraints (S1)–(S4). The relative importance of the different soft constraints is set by a weight factor for each soft constraint. Since the weights will be of no relevance to our arguments later on, we assume that appropriate weighting has been applied within $c_{S1}, \ldots, c_{S4}$. A detailed specification of the penalty functions can be found in [10].

Definition 1 (Curriculum-based Course Timetabling Problem) Given a CB-CTT instance I, find a feasible timetable τ such that $c(\tau)$ is minimal.

A max-min fair variant of the CB-CTT problem was defined in [26]. Given a CB-CTT instance with curricula $u_1, \ldots, u_k$. The allocation vector of a timetable τ is given by:

$$A(\tau) = (c(u_1, \tau), c(u_2, \tau), \ldots, c(u_k, \tau)) \ , \tag{2}$$

where $c(u_j, \tau) = \sum_{1 \leq i \leq 4} c_{Si}(u_j, \tau)$, $i \in \{1, \ldots, 4\}$, is the CB-CTT objective function restricted to the events of the courses in curriculum u_j, $j \in \{1, \ldots, k\}$.

Definition 2 (Max-min Fair Curriculum-based Course Timetabling Problem) Given a CB-CTT instance I, find a feasible timetable τ such that $A(\tau)$ is max-min fair.

2.3 The Assignment Problem

The assignment problem is a classical problem in combinatorial optimization which appears in many applications, for example personnel scheduling, job scheduling and object tracking, just to name a few. For a comprehensive overview of the body of research on the assignment problem and the applications see [5,30]. In CB-CTT problem for example, the assignment problem appears as a subproblem [21,24]. There exist polynomial-time algorithms for many variants of the assignment problem.

Let A and B be two sets of cardinality n. An assignment of the elements of A to the elements of B is a bijection $\sigma : A \to B$. Typically, assignment problems are optimization problems, i. e., among all bijections from A to B, we are looking for one that is optimal with respect to a certain objective function. In the context of (fair) curriculum-based course timetabling, we are in particular interested in two variants of the assignment problem, namely the linear assignment problem (LSAP) and the lexicographic bottleneck assignment problem (LBAP).

Definition 3 (Linear Sum Assignment Problem (LSAP)) Given a cost function $c : A \times B \to \mathbb{R}$, find a bijection $\sigma : A \to B$ such that $\sum_{a \in A} c(a, \sigma(a))$ is minimal.

There exist various algorithms for solving LSAPs efficiently, including the well-known Hungarian algorithm [29, p. 248ff] and network flow algorithms [13]. In the following, let $T_{\text{LSAP}}(n)$ be the time complexity of solving an LSAP instance with $|A| = |B| = n$.

When solving MMF-CB-CTT problems using the decomposition proposed in the next section, the task of finding max-min fair assignments occurs as a subproblem. An assignment $\sigma : A \to B$ is called max-min fair, if for any assignment $\sigma' : A \to B$ we have $\mathbf{c}(\sigma) \preceq \mathbf{c}(\sigma')$, where $\mathbf{c}(\sigma) = (c(a, \sigma(a)))_{a \in A}$ and $\preceq$ is the max-min fair comparison from Section 2.1.

Definition 4 (Lexicographic Bottleneck Assignment Problem (LBAP)) Given a cost function $c : A \times B \to \mathbb{R}$, find a max-min fair bijection $\sigma : A \to B$.

A LBAP can be transformed into a LSAP by scaling the cost values appropriately. This results in an exponential blow-up of the cost values, which may be undesirable in practical applications [4]. Alternatively, an LBAP can be reduced to a *linear vector assignment problem*, which belongs to the class of *algebraic assignment problems* [6]. Using this reduction, a given LBAP with a cost function c can be solved in time $O(kn^3)$, where k is the number of distinct values attained by c [9]. The reduction is straightforward: For each $i \in \{1, \ldots, k\}$, let $\mathbf{e}_i \in \mathbb{N}^k$, be a vector whose i-th component is 1 and all other components are 0. The cost function c is replaced by a vector-valued function $c' : A \times B \to \mathbb{N}^k$ such that $c'(a, \sigma(a)) = \mathbf{e}_i$ if $c(a, \sigma(a))$ is the i-th largest value attained by c. An assignment σ that yields a lexicographically minimal cost vector $\sum_{a \in A} c'(a, \sigma(a))$ is an optimal solution to the corresponding LBAP.

3 Problem Decomposition

It is a common approach to decompose the CB-CTT problem in a way that room and timeslot assignment is preformed separately, see for example [21,24]. For courses in a single timeslot, an optimal room assignment can be determined by solving a LSAP instance. In this section, we establish a similar result for the MMF-CB-CTT problem: An optimal room assignment for the courses in a single timeslot can be determined by solving a (generalized) LBAP instance. In the spirit of Benders decomposition approach [2], we further consider decompositions of sum optimization problems into master and subproblems such that the subproblems can be solved efficiently. We derive sufficient conditions under which such a decomposition of a sum optimization problem carries over to its lexicographic bottleneck counterpart.

Let the sum optimization problem (SOP) be the following combinatorial problem:

$$\begin{aligned} \text{(SOP)} \quad & \min c \cdot x \\ & \text{s.t. } x \in X \ , \end{aligned}$$

where $c \in \mathbb{N}^m$ and $X \subseteq \{0,1\}^m$ and X contains all feasible assignments of the variables $x_1, \ldots, x_m$. For the LSAP from Section 2.3 for example, X encodes the bijections $A \to B$. Let T_X be the time required to solve the SOP on the combinatorial structure X. Each elementary operation (e.g., addition) in the solution procedure is assumed to take constant time. Now consider the following generalized lexicographic bottleneck optimization problem (GLBOP):

$$\begin{aligned} \text{(GLBOP)} \quad & \min_{\preceq} f(x) = (w_1 x_{i_1}, \ldots, w_\ell x_{i_\ell}) \\ & \text{s.t. } x \in X \ , \end{aligned} \tag{3}$$

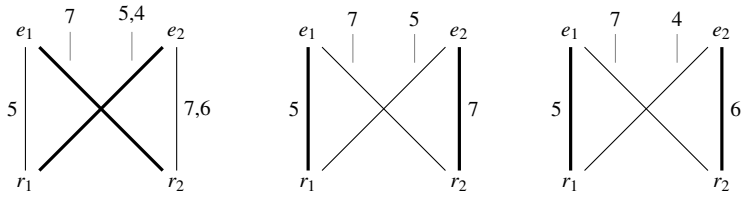
where $f : X \to \mathbb{N}^\ell$ for some $\ell \in \mathbb{N}$, $w_1, \ldots, w_\ell \in \mathbb{N}$ are called weights and $i_1, \ldots, i_\ell \in \{1, \ldots, m\}$. The minimum is taken with respect to the relation $\preceq$ from Section 2.1. Please note that this is a generalization of the definition of the LBOP in [4,9] since each of the variables can appear in several components of f.

Theorem 1 *A GLBOP instance with ℓ weights, t of them distinct, can be solved in time $O(\ell^2 + t \cdot T_X)$.*

Proof Following the vectorial approach of Della Croce et al. in [9], we reduce the GLBOP to a vectorial sum optimization problem (VSOP). We construct a cost matrix $Z \in \mathbb{N}^{m \times t}$, which contains in each row i a cost vector for the variable x_i. With each weight w we associate a cost vector $(d_1, \ldots, d_t) \in \mathbb{N}^t$ such that $d_i = 1$, if w is the i-th largest weight, and $d_i = 0$ otherwise. Let the i-th row of the cost matrix Z be the sum of all cost vectors associated to the weights that occur in a product with x_i in any component of f. The VSOP has the following form:

$$\begin{aligned} \min_{\preceq_{lex}} \quad & \tilde{f}(x) = x \cdot Z \\ & \text{s.t. } x \in X \ . \end{aligned} \tag{4}$$

Suppose for a contradiction that an optimal solution x^* to the VSOP (4) is not optimal for the GLBOP (3). Then there is a solution $x' \in X$ such that $f(x') \prec f(x^*)$. As a consequence, $\tilde{f}(x') \prec_{lex} \tilde{f}(x^*)$ due to the construction of the cost vectors above. This is a contradiction to



(a) Room assignment problem: optimal cost 7, 5, 4

(b) LBAP 1: optimal cost 7,5

(c) LBAP 2: optimal cost 6,5

Fig. 1: A room assignment problem examples with two courses, e_1 and e_2, and two rooms r_1 and r_2. None of the two LBAPs (b) and (c) models the room assignment problem (a): The optimal solutions highlighted for LBAPs are not optimal for the room assignment problem.

the optimality of x^*. The VSOP (4) can be solved in time $O(t \cdot T_X)$ because each elementary operation involving the weights is now performed on a vector of length t. The time complexity of the reduction is $O(\ell^2)$.

A similar result can be obtained by modifying the cost scaling approach in [4]: Taking the sum of the scaled weights that occur in a product involving the variable x_i is equivalent to adding the corresponding cost vectors during the construction of the cost matrix Z in the proof of Theorem 1. This approach results in a direct reduction of the GLBOP to the SOP with weights that are exponentially large compared to the original weights.

We will now apply the result above to a decomposition of the MMF-CB-CTT problem. The decomposition isolates, for a single period, the assignment of courses to rooms, from the rest of the problem. So, the task is to find an optimal room assignment for a given period assuming the rest of the timetable is fixed. Optimizing the rest of the timetable can be considered the master problem. For the CB-CTT problem the room assignment subproblem is a LSAP and hence can be solved efficiently. Our goal is to show that the room assignment subproblem of the MMF-CB-CTT problem can be solved efficiently as well. In the following, let C be the set of courses, R be the set of rooms, P be the set of periods, and $U \subseteq \mathscr{P}(C)$ be the curricula. By C_p we denote the set of courses scheduled in the period $p \in P$. Please note that C_p is determined by the solution of the master problem. Furthermore, let $U_e = \{u \in U \mid e \in u\}$.

Figure 1 illustrates why simply solving a LBAP does not account for the intricacies of the room assignment subproblem of the MMF-CB-CTT problem. To keep things simple, there is only a single period p, two courses $C = C_p = \{e_1, e_2\}$, and two rooms $R = \{r_1, r_2\}$. However, the course e_2 is in two curricula and thus determines two entries of the overall allocation vector. The cost on each edge connected to e_1 shows the costs generated for each of the two curricula by assigning a particular room to e_2. Figures 1b and 1c are LBAP instances that reflect only costs for one of the two curricula of e_2. The assignments highlighted are both optimal solutions to the individual LBAPs, but none of them is optimal for the room assignment shown in Figure 1a.

Theorem 2 *The room assignment subproblem of the MMF-CB-CTT problem is a GLBOP.*

Proof The room assignment subproblem of the CB-CTT problem, for a fixed period p, is a LSAP. Therefore, restricting the set of courses to a single curriculum $u \in U$, also yields an LSAP as room assignment subproblem, however, with at most one course of u in C_p

(due to the conflict constraints). Thus, assigning a course e to a room r in p completes the timetable τ from the perspective of all curricula in U_e and therefore determines their cost entries $w_{u,e,r} = c(u,\tau)$ for each $u \in U_e$ in the allocation vector (2). Thus, the room assignment subproblem can be written as

$$\begin{aligned} \min_{\preceq} \quad & \prod_{e,r} (w_{u,e,r} \cdot x_{e,r})_{u \in U_e} \\ \text{s. t. } & x \in X \quad , \end{aligned} \tag{5}$$

where $\prod$ denotes the concatenation of tuples and X contains all valid room assignments of the courses in period p to the rooms R. The variables are $\{x_{e,r}\}_{e \in C_p, r \in R}$ and $x_{e,r} = 1$ if course e is assigned to room r in period p, and 0 otherwise. X contains all feasible variable assignments. Clearly, problem (5) is a GLBOP.

Corollary 1 *For a period p, the room assignment subproblem* (5) *with* $n = |C_p|$ *courses can be solved in time* $O(|U|^2 + |U| \cdot T_{\text{LSAP}}(n))$.

Proof Problem (5) is an assignment problem, just as the room assignment subproblem of the CB-CTT problem. Only the objective function is different. Hence, combining Theorems 1 and 2 yields the result.

Remark 1 As noted by Lach and Lübbecke in [21], the room stability constraint (S4) introduces dependencies between room assignments in different timeslots, which prevents us from extending the decomposition to more than a single period.

Remark 2 In the general case, whenever there is a decomposition of a SOP into a master problem and a subproblem which can be solved efficiently, the subproblem of the corresponding LBOP can be solved efficiently if it is a GLBOP. Since complex SOPs are much better studied than LBOPs this observation may be useful when adding max-min fairness objectives to existing SOPs.

In Section 5 we will provide experimental evidence that solving the room assignment subproblem to optimality is useful for improving the performance of a heuristic algorithm for the MMF-CB-CTT problem.

4 Quantifying Max-min Fairness

When dealing with randomized optimization algorithms, one can employ a wealth of statistical tools to extract meaningful information about algorithms' absolute and relative performance. These tools include statistical tests such as the Wilcoxon rank-sum test and measures such as the mean quality of the solutions, the standard deviation, the median quality, the quality of the best and worst solutions, and so on. Due to the qualitative nature of max-min fairness, so far only statistical tools based on ranking can be used for evaluating randomized max-min fair optimization algorithms. In this section we propose a novel approach to partially overcome this limitation. We consider max-min fair minimization problems whose solutions induce allocation vectors in $\mathbb{N}_0^n$. The main idea is to construct an isomorphism from the set of sorted allocation vectors to an interval in the natural numbers ordered by the usual $\leq$ relation. Using this isomorphism, we can perform all operations on natural/real numbers, round the result to the nearest integer, and retrieve the corresponding allocation vector. This means that if we have a set of allocation vectors, we can determine for example an allocation

vector close to the average allocation. Thus, in our experiments in the next section we will be able to compare the average solution quality of two algorithms for the MMF-CB-CTT problems.

Lemma 1 *Let* $\mathbb{A}_n = \{(x_1,\ldots,x_n) \in \mathbb{N}_0^n \mid x_1 \geq x_2 \geq \ldots \geq x_n\}$. *Let the function* $\text{rank} : \mathbb{A}_n \to \mathbb{N}_0$ *be*

$$\text{rank}(x_1,\ldots,x_n) = \sum_{i=1}^{n} \binom{n+x_i-i}{x_i-1} \ , \tag{6}$$

where $\binom{k}{-1} = 0$ *for all* $k \in \mathbb{N}_0$. *Then* $\text{rank} : (\mathbb{A}_n, \preceq_{lex}) \to (\mathbb{N}_0, \leq)$ *is an isomorphism.*

Proof $(\mathbb{A}_n, \preceq_{lex})$ is a linearly ordered set with least element $(0,\ldots,0)$. Therefore, for each $X \in \mathbb{A}_n$ there is a unique natural number $r_X = |\{Y \in \mathbb{A}_n \mid Y \prec_{lex} X\}|$. Thus, the function mapping each $X \in \mathbb{A}_n$ to r_X is a bijective mapping and it is order-preserving as required. It remains to be shown that $\text{rank}(X)$ computes r_X for all $X \in \mathbb{A}_n$.

Let $X = (x_1,\ldots,x_n) \in \mathbb{A}_n$. The value r_X can be determined by the following recursion:

$$r_{x_1,\ldots,x_n} = r_{x_2,\ldots,x_n} + r_{x_1,0,\ldots,0} \ . \tag{7}$$

This recursion separately counts the non-increasing sequences $\{Y \in \mathbb{A}_n \mid (x_1,0,\ldots,0) \preceq_{lex} Y \prec_{lex} X\}$ and $\{Y \in \mathbb{A}_n \mid Y \prec_{lex} (x_1,0,\ldots,0)\}$. The number of sorted sequences of length n over an ordered alphabet of size k is $\binom{n+k}{k}$. Thus,

$$r_{\underbrace{x_1,0,\ldots,0}_{n}} = \binom{n+x_1-1}{x_1-1} \ .$$

In particular, for $x \in \mathbb{N}_0$ we have $r_x = \binom{x}{x-1} = x$. Unfolding the recursion (7) yields (6). Therefore, $\text{rank}(X)$ computes r_X.

The relation $\preceq$ gives rise to an equivalence relation $\sim$ on $\mathbb{N}_0^n$, the set of allocation vectors of dimension n: Two allocation vectors X and Y are equivalent if $X \preceq Y$ and $Y \preceq X$. We denote by $[X]_\sim$ the equivalence class containing X.

Theorem 3 *Let* $X \in \mathbb{N}_0^n$. *Then the function*

$$\begin{aligned} \rho : (\mathbb{N}_0{}^n_{/\sim}, \preceq) &\to (\mathbb{N}_0, \leq) \\ [X]_\sim &\mapsto \text{rank}(\overleftarrow{X}) \ , \end{aligned}$$

is an isomorphism.

Proof For each equivalence class, we can choose as a representative the unique allocation whose elements occur in non-increasing order, and use the isomorphism rank from Lemma 1.

We will now consider maximization problems and the corresponding max-min fair comparison $\preceq_{max}$. In a similar fashion to $\preceq$, it gives rise to an equivalence relation $\sim_{max}$ on $\mathbb{N}_0^n$. We will show that, if there is a certain maximum amount resources for all stakeholders, then we can use the function rank again to construct an isomorphism between allocation vectors and a subset of the natural numbers. The assumption of a maximum amount of resources is necessary, since, if the resource allocated to a stakeholder are unbounded, the number of fairer and unfairer allocations are infinite. The isomorphism is useful in order to measure the quality and the quality difference of allocation vectors for maximization problems.

Theorem 4 *Let $m,n \in \mathbb{N}_0$, $D = \{0,\ldots,m\}^n$, $M = (m,\ldots,m) \in D$, and $I = \{\text{rank}(M - \vec{X}) \mid X \in D\}$. Then the function*

$$\begin{aligned} \rho' : (D_{/\sim_{max}}, \preceq_{max}) &\to (I, \leq) \\ [X]_{\sim_{max}} &\mapsto \text{rank}(M - \vec{X}) \quad , \end{aligned}$$

is an isomorphism. □

5 Evaluation

In this section we are going to present experimental evidence for the usefulness decomposition presented in Section 3. We compare the performance of two randomized heuristic algorithms for the MMF-CB-CTT problem, both of which are based on the algorithm MAXMINFAIR_SA from [26]. The first algorithm is the one that performed best in [26]. It uses a decomposition of the MMF-CB-CTT problem that is similar to the one presented in this work, but models the room assignment subproblem as LSAP. Thus, we will refer to this algorithm by MMF_SA_LSAP. The second algorithm, MMF_SA_GLBOP, uses the MMF-CB-CTT decomposition from Section 3 and thus solves a GLBOP to obtain an optimal room assignment for a given period. Apart from how the room assignment subproblem is solved, the the two algorithms are identical. We compare both algorithms with respect to the best solutions they produce as well as the average solution quality per instance. In order to determine average allocations, we use the isomorphism ρ from Theorem 3 as described in the previous section. Our results indicate that the algorithm which makes GLBOP room assignment significantly outperforms the other one.

The algorithm MAXMINFAIR_SA is a variant of simulated annealing (SA) [17] which has been tailored to the MMF-CB-CTT problem. Simulated annealing iteratively generates new candidate solutions and keeps (or *accepts*) a new solution if it is better. If the new solution is worse, then it is accepted with a certain probability which depends on the temperature ϑ. There are three crucial design choices when adapting simulated annealing to a particular problem: The cooling schedule, the acceptance criterion, and the neighborhood structure. In both algorithms under consideration, MMF_SA_GLBOP and MMF_SA_LSAP, we use the standard geometric cooling schedule, which lets the temperature decay exponentially from a given ϑ_{max} to a given ϑ_{min}. Both algorithms use the acceptance criterion based on the component-wise energy difference, which performed best in a comparison of different acceptance criteria in [26]. Also, both algorithms use a neighborhood structure based on the well-known Kempe-move. A Kempe-move swaps a subset of the events assigned to two given periods such that the conflict constraints between the events are not violated. The difference between both algorithms is how the room-assignment subproblems are solved after performing a Kempe-move: MMF_SA_LSAP solves two LSAPs in order to assign rooms efficiently, while MMF_SA_GLBOP solves two GLBOPs. Thus, the second room assignment performed by MMF_SA_GLBOP is optimal with respect to the rest of the timetable. Our evaluation shows that this is beneficial for the overall algorithm performance.

In order to compare the two algorithms, we performed 50 independent runs for each algorithm on the 21 CB-CTT instances from track three of the timetabling competition ITC2007 [10]. In each run, 1000000 iterations of the simulated annealing procedure were performed. We did not tweak the temperature-related parameters of the algorithms extensively, but determined experimentally that $\vartheta_{max} = 5$ and $\vartheta_{min} = 0.01$, as suggested in [26] work well. Table 1 shows the best allocation vectors obtained by MMF_SA_LSAP and

MMF_SA_GLBOP on the 21 CB-CTT instances. To aid the presentation of the results, Table 1 shows the allocation vectors in a compressed form: The penalty values are sorted in non-increasing orders and repetitions of values are shown as exponents. For example, a table entry of $5^2 10^1 1$ denotes an allocation vector in which the penalty value 5 appears two times, 1 appears one and 0 eleven times. Note that from the max-min fairness perspective it is not important to which curriculum a penalty value corresponds, so we omit this information. The results in Table 1 show that MMF_SA_GLBOP finds better solutions than MMF_SA_LSAP on 18 instances, while the best solutions found by MMF_SA_LSAP are better on `comp04` and `comp18`. In addition to the best allocation vectors, Table 1 shows the average allocations over the 50 runs for each instance. The average allocations have been computed using ρ from Section 4: The allocation vectors were mapped to the natural numbers, then the average was calculated and rounded to the nearest integer. Finally, the result was mapped back to the corresponding equivalence class of allocation vectors. A comparison of the average allocations shows that in this respect, MMF_SA_GLBOP outperforms MMF_SA_LSAP on 16 instances while it is beaten on the instances `comp05`, `comp08`, `comp15` and `comp21`.

We also performed the one-sided Wilcoxon rank-sum test with a significance level of 0.01. According to the test, MMF_SA_GLBOP yields significantly better results than MMF_SA_LSAP on instances `comp02`, `comp06`, `comp07`, `comp08`, `comp10`, `comp13`, `comp14`, `comp17`, `comp18`, `comp19`, `comp20` and `comp21`. In contrast, MMF_SA_LSAP is not significantly better than MMF_SA_GLBOP on any of the 21 instances with this significance level. The results of the Wilcoxon test are consistent with the data in Table 1.

In contrast to experimental setup in [26], we did not use a timeout, but set a fixed number of iterations for the direct comparison of MMF_SA_GLBOP and MMF_SA_LSAP. The reason for this decision is that in practice, solving GLBOP instances takes significantly more time than solving LSAPs. The increase in runtime is due to overhead required by construction of the cost matrix ($O(|U|^2)$) and the $O(|U|)$ factor for solving the linear vector assignment problem (see Corollary 1). Since we are mainly interested in the implications of modelling the room assignment subproblem with a GLBOP instead of an LSAP, both algorithms should be able to solve a similar number of room-assignment subproblems. From the data shown in Table 1 we can conclude, that using the decomposition presented in Section 3 is clearly the smarter choice, since, after solving equally many subproblems, it produces superior results compared to the approach from [26]. However, if we employ the timeout as it was required for the ITC2007 competition (see [10]), MMF_SA_LSAP is the better choice, because it can perform significantly more iterations within the given timeout.

6 Conclusion

In this work we proposed a decomposition of the MMF-CB-CT problem from [26]. The decomposition models the room assignment subproblem as a assignment problem with a generalized lexicographic bottleneck objective. The decomposition enables us to compute an optimal room assignment for a given period in polynomial time. We use this result to improve the performance of the MaxMinFair_SA algorithm proposed in [26], which originally modelled the room assignment subproblem as an LSAP. An optimal solution to the LSAP however is not necessarily optimal for the room assignment subproblem of the MMF-CB-CTT problem. In our experiments we compare the performance of two variants of the MaxMinFair_SA algorithm, wich differ only with respect to how the room assignment subproblem is tackled. Our results indicate that solving the room assignment subproblem

Table 1: Comparison of the best and average allocation vectors of the solutions found by MMF_SA_LSAP and MMF_SA_GLBOP on the 21 CB-CTT instances from [25] for 50 independent runs per instance and per algorithm. For each instance, the best results and best average results are marked in bold face.

	MMF_SA_GLBOP		MMF_SA_LSAP	
Instance	best	average	best	average
comp01.ectt	$\mathbf{5^2,0^{12}}$	$\mathbf{6,5,4^5,3^4,2,1,0}$	$5^2,1,0^{11}$	$6,5^3,4,2^2,1^5,0^2$
comp02.ectt	$\mathbf{2^{32},1^5,0^{33}}$	$\mathbf{4^5,3^{18},2^{16},1^{18},0^{13}}$	$4,3^3,2^{26},1^{12},0^{28}$	$5^2,4^{30},3^4,2^{17},1^{16},0$
comp03.ectt	$\mathbf{6^4,4^{11},2^{23},1,0^{29}}$	$\mathbf{6^5,5^{40},4^8,3^2,2^4,0^9}$	$6^4,4^{12},2^{27},1^3,0^{22}$	$6^{10},5^9,4,3^{24},2,1^3,0^{20}$
comp04.ectt	$6^4,4^2,2^4,1^7,0^{40}$	$\mathbf{6^4,4^3,3^6,2^{34},1^8,0^2}$	$\mathbf{6^4,4^2,2^4,0^{47}}$	$6^4,4^3,3^{11},2^{33},1^4,0^2$
comp05.ectt	$\mathbf{19^2,18^3,17^3,16^5,15^2,14^8},\ldots$	$20^3,18^6,16,15^9,14^2,13^3,\ldots$	$19^2,18^3,17^3,16^6,15^2,14^7,\ldots$	$\mathbf{20^2,19^4,18^{11},16^3,15^{30},14^7},\ldots$
comp06.ectt	$\mathbf{12,4,2^{29},1^{15},0^{24}}$	$\mathbf{12,5^6,4^2,3^9,2^{43},1^4,0^5}$	$12,4^3,3,2^{31},1^{12},0^{22}$	$12,5^{11},4^3,3^5,2^4,1^5,0^{41}$
comp07.ectt	$\mathbf{6,2^{14},1^{18},0^{44}}$	$\mathbf{6,4,3^{21},2^9,1^{15},0^{30}}$	$6,3,2^{30},1^{25},0^{20}$	$6,4^2,3^{22},2^7,1^{11},0^{34}$
comp08.ectt	$\mathbf{6^4,4^2,2^9,1^6,0^{40}}$	$6^4,4^3,3^{23},2^{15},1^{14},0^2$	$6^4,4^2,2^{11},1^4,0^{40}$	$\mathbf{6^4,4^4,3^3,2^9,1^{20},0^{21}}$
comp09.ectt	$\mathbf{6^9,4^{11},2^{18},1^2,0^{35}}$	$\mathbf{6^{10},5^3,4^8,3^{21},2^{12},1^{13},0^8}$	$6^9,4^{14},2^{12},1^6,0^{34}$	$6^{10},5^5,4^4,3^{10},2^{10},1^{17},0^{19}$
comp10.ectt	$\mathbf{2^{12},1^5,0^{50}}$	$\mathbf{4,3^{34},2^{13},0^{19}}$	$2^{20},1^9,0^{38}$	$4^{10},3^4,2^{24},1^6,0^{23}$
comp11.ectt	0^{13}	0^{13}	0^{13}	0^{13}
comp12.ectt	$\mathbf{10^4,8^{26},7,6^{50},5^2,4^{38}},\ldots$	$\mathbf{12^2,10^2,9^{45},8^2,7^{12},6^{11}},\ldots$	$10^5,8^{26},6^{48},5^6,4^{42},3^4,\ldots$	$12^3,11^{19},10^9,9^2,8^{21},7^{12},\ldots$
comp13.ectt	$\mathbf{6^6,4^4,2^{12},1^2,0^{42}}$	$\mathbf{6^6,4^6,3^{11},2^{24},1^6,0^{13}}$	$6^6,4^4,2^{21},1^4,0^{31}$	$6^6,4^7,3^{11},2,1^{29},0^{12}$
comp14.ectt	$\mathbf{8^4,4^3,2^{13},1^3,0^{37}}$	$\mathbf{8^4,4^5,3^5,2^{32},1^{11},0^3}$	$8^4,4^3,2^{16},1^2,0^{35}$	$8^4,4^{13},3^{22},2^{10},1^7,0^4$
comp15.ectt	$\mathbf{6^4,4^{10},2^{22},1^4,0^{28}}$	$6^{11},5^3,4^8,3^{13},2^7,1^2,0^{24}$	$6^4,4^{11},3,2^{25},1^5,0^{22}$	$\mathbf{6^8,5^3,4^{18},3,2^{12},1^{18},0^8}$
comp16.ectt	$\mathbf{4^5,2^{14},1^3,0^{49}}$	$\mathbf{5^5,4^{10},3^6,2^{19},1^6,0^{25}}$	$4^5,3,2^{17},1^7,0^{41}$	$5^{12},4^4,3^{16},2^4,1^{22},0^{13}$
comp17.ectt	$\mathbf{10^2,6^2,4^5,2^{31},1^8,0^{22}}$	$\mathbf{10^2,6^2,5,4^{32},2^6,0^{27}}$	$10^2,6^2,4^8,3,2^{25},1^{12},0^{20}$	$10^2,6^2,5^9,4^{29},3^{16},2^7,1^5$
comp18.ectt	$6,4^{18},3,2^{15},1^4,0^{13}$	$\mathbf{7,6^{19},5^6,4^3,3^2,2^9,1^{11},0}$	$\mathbf{6,4^{17},3^3,2^{14},1^3,0^{14}}$	$7^4,5^5,4^{20},3^9,2^4,1^9,0$
comp19.ectt	$\mathbf{6^4,4^6,2^{12},1^5,0^{39}}$	$\mathbf{6^{14},5^9,4^7,2^{18},1^6,0^{12}}$	$6^4,4^7,2^{14},1^{10},0^{31}$	$7^2,6^4,5^{16},4^2,3^{19},2^4,1^{16},0^3$
comp20.ectt	$\mathbf{4^3,3^2,2^{27},1^{16},0^{30}}$	$\mathbf{5^2,4^{24},3^4,2,1,0^{46}}$	$4^5,3^2,2^{34},1^{13},0^{24}$	$5^{16},4^6,3^2,1^7,0^{47}$
comp21.ectt	$\mathbf{10,6^4,4^{15},3,2^{25},1^7,0^{25}}$	$10,6^{10},5^{33},4^4,2,1^6,0^{23}$	$10,6^4,5,4^{16},2^{27},1^4,0^{25}$	$\mathbf{10,6^9,5^{15},4^6,3,2^{31},0^{15}}$

as proposed in Section 3 improves the performance of the MAXMINFAIR_SA algorithm on most of the ITC2007 benchmark instances. Furthermore, we proposed a measure for quantifying how fair a timetable is with respect to max-min fairness. Using this measure helps to apply statistical methods in the analysis of the performance of randomized optimization algorithms for optimization problems with a bottleneck objective. In particular, it enables us to compare the average solution quality of the two variants of the MAXMINFAIR_SA algorithm.

References

1. Ruggero Bellio, Luca DiGaspero, and Andrea Schaerf. Design and statistical analysis of a hybrid local search algorithm for course timetabling. *Journal of Scheduling*, 15:49–61, 2012.
2. J. F. Benders. Partitioning procedures for solving mixed variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.
3. Dimitris Bertsimas, Vivek F. Farias, and Nikolaos Trichakis. The price of fairness. *Operations Research*, 59(1):17–31, 2011.
4. R. E. Burkard and F. Rendl. Lexicographic bottleneck problems. *Operations Research Letters*, 10:303–308, 1991.
5. Rainer Burkard, Mauro Dell'Amico, and Silvano Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.
6. Rainer Ernst Burkard and Uwe Zimmermann. *Weakly admissible transformations for solving algebraic assignment and transportation problems*, volume 12 of *Mathematical Progamming Study*, chapter 1, pages 1–18. North-Holland, 1980.
7. Edmund K. Burke, Jakub Marecek, Andrew J. Parkes, and Hana Rudová. Decomposition, reformulation, and diving in university course timetabling. *Computers & OR*, 37(3):582–597, 2010.
8. Edmund K. Burke, Jakub Mareček, Andrew J. Parkes, and Hana Rudová. A branch-and-cut procedure for the Udine course timetabling problem. *Annals of Operations Research*, 194(1):71–87, 2011.
9. F. Della Croce, V. Th. Paschos, and A. Tsoukias. An improved general procedure for lexicographic bottleneck problems. *Operations Research Letters*, 24:187–194, 1999.
10. Luca Di Gaspero, Barry McCollum, and Andrea Schaerf. The second international timetabling competition (ITC-2007): Curriculum-based course timetabling (Track 3). Technical Report QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0/1, School of Electronics, Electrical Engineering and Computer Science, Queens University, Belfast (UK), August 2007.
11. A.T Ernst, H Jiang, M Krishnamoorthy, and D Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3 – 27, 2004.
12. Allan M. Feldman and Roberto Serrano. *Welfare economics and social choice theory*. Springer, New York, NY, 2nd edition, 2006.
13. H. Gabow and R. Tarjan. Faster scaling algorithms for network problems. *SIAM Journal on Computing*, 18(5):1013–1036, 1989.
14. Iiro Harjunkoski and Ignacio E. Grossmann. Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Computers & Chemical Engineering*, 26(11):1533 – 1552, 2002.
15. Rajendra K. Jain, Dah-Ming W. Chiu, and William R. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Technical report, Digital Equipment Corporation, September 1984.
16. Stacy L. Janak, Christodoulos A. Floudas, Josef Kallrath, and Norbert Vormbrock. Production scheduling of a large-scale industrial batch plant. i. short-term and medium-term scheduling. *Industrial & Engineering Chemistry Research*, 45(25):8234–8252, 2006.
17. Scott Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
18. Jon Kleinberg, Yuval Rabani, and Éva Tardos. Fairness in routing and load balancing. *Journal of Computer and System Sciences*, 63(1):2–20, 2001.
19. Donald E. Knuth. *The Art of Computer Programming, Volume III: Sorting and Searching*. Addison-Wesley, 1973.
20. A. Kumar and J. Kleinberg. Fairness measures for resource allocation. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 75–85, 2000.
21. Gerald Lach and Marco E. Lübbecke. Curriculum based course timetabling: new solutions to Udine benchmark instances. *Annals of Operations Research*, 194:255–272, 2012.

22. Tian Lan, David Kao, Mung Chiang, and Ashutosh Sabharwal. An axiomatic theory of fairness in network resource allocation. In *INFOCOM*, pages 1343–1351. IEEE, 2010.
23. Zhipeng Lü and Jin-Kao Hao. Adaptive tabu search for course timetabling. *European Journal of Operational Research*, 200(1):235–244, 2010.
24. Zhipeng Lü and Jin-Kao Hao. Adaptive Tabu Search for course timetabling. *European Journal of Operational Research*, 200(1):235–244, 2010.
25. Barry McCollum, Andrea Schaerf, Ben Paechter, Paul McMullan, Rhyd Lewis, Andrew J. Parkes, Luca Di Gaspero, Rong Qu, and Edmund K. Burke. Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing*, 22:120–130, 2010.
26. Moritz Mühlenthaler and Rolf Wanka. Fairness in academic course timetabling. In *Proc. 9th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT)*, pages 114–130, 2012.
27. Tomáš Müller. ITC2007 solver description: A hybrid approach. *Annals of Operations Research*, 172(1):429–446, 2009.
28. Włodzimierz Ogryczak. Bicriteria models for fair and efficient resource allocation. In *Proc. 2nd International Conference on Social Informatics (SocInfo)*, pages 140–159. Springer, 2010.
29. Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization; Algorithms and Complexity*. Dover Publications, 1998.
30. David W. Pentico. Assignment problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774 – 793, 2007.
31. Celso C. Ribeiro and Sebastián Urrutia. Scheduling the brazilian soccer tournament with fairness and broadcast objectives. In *Proc. 6th Int. Conf. on Practice and Theory of Automated Timetabling*, PATAT, pages 147–157, Berlin, Heidelberg, 2007. Springer-Verlag.
32. Tomáaš Müller and Hana Rudová. Real-life curriculum-based timetabling. In *Proc. 9th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT)*, pages 57–72, 2012.
33. Pieter Smet, Simon Martin, Djamila Ouelhadj, Ender Özcan, and Greet Vanden Berghe. Investigation of fairness measures for nurse rostering. In *Proc. 9th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT)*, pages 369–372, 2012.
34. M. J. Soomer and G. M. Koole. Fairness in the aircraft landing problem. In *Proceedings of the Anna Valicek Competition 2008*, 2008.
35. Pamela H. Vance, Cynthia Barnhart, Ellis L. Johnson, and George L. Nemhauser. Airline Crew Scheduling: A new formulation and decomposition algorithm. *Operations Research*, 45(2), M 1997.