PRODUCTION AND OPERATIONS MANAGEMENT POMS
Vol. 30, No. 10, October 2021, pp. 3540–3559
ISSN 1059-1478 | EISSN 1937-5956 | 21 | 3010 | 3540
DOI 10.1111/poms.13449

Randomized Scheduling Mechanisms: Assigning Course Seats in a Fair and Efficient Way

Martin Bichler*, Soeren Merting
Department of Computer Science, Technical University of Munich, Boltzmannstr. 3, Garching, 85748, Germany, bichler@in.tum.de, merting@in.tum.de

Course assignment is a very widespread problem in education and beyond. Typically, students have preferences for bundles of course seats or course schedules over the week, but courses have limited capacity. This is an interesting and frequent application of distributed scheduling, where payments cannot be used to implement the efficient allocation. First-Come First-Served (FCFS) is simple and the most widely used assignment rule in practice, but it leads to inefficient outcomes and envy in the allocation. It was recently shown that randomized economic mechanisms that do not require monetary transfers can have attractive economic and computational properties, which were considered incompatible for deterministic alternatives. We use a mixed-methods design including field and laboratory experiments, a survey, and simulations to analyze such randomized mechanisms empirically. Implementing randomized scheduling in the field also required us to develop a solution to a new preference elicitation problem that is central to these mechanisms. The results of our empirical work shed light on the advantages that randomized scheduling mechanisms have over FCFS in the field, but also on the challenges. The resulting course assignment system was adopted permanently and is now used to solve course assignment problems with more than 1700 students every year.

Key words: distributed scheduling; course assignment; education; optimization; randomization
History: Received: August 2019; Accepted: April 2021 by Subodha Kumar, after 2 revisions.
*Corresponding author.

1. Introduction

Scheduling is ubiquitous in organizations and often involves the assignment of people with preferences to scarce resources or tasks. Examples include the allocation of surgeons to hospital operating rooms, workers to shifts or projects over time, or students to different courses on a weekly schedule. There is a huge literature on optimization formulations and decision support tools for rostering, staffing, or scheduling, which typically analyzes such problems from the point of view of a single decision maker or planner (Sampson 2004, May et al. 2011, Jung et al. 2019, Ou et al. 2010). Kreipl and Pinedo (2004) provide an excellent overview of the variety of scheduling problems that arises in and across organizations. For most of these problems, peoples' private preferences matter. Surgeons have preferences for certain time slots for their operations (May et al. 2011), and workers have preferences for times of the day and free time (Burke et al. 2004). However, almost all of the scheduling literature assumes complete information about the participants' preferences. The focus is on the mathematical solution of the resulting optimization problem, given that the true preferences are available. This might be a too strong assumption for most applications, but leads to a fundamental question: How can participants be incentivized to reveal their preferences truthfully?

Mechanism design investigates economic mechanisms with a focus on incentive-compatibility. A mechanism is *incentive-compatible* if every participant can achieve the best outcome to themselves just by acting according to their true preferences. If participants have dominant strategies to reveal preferences truthfully, we talk about *strategy-proofness*. Auction designs such as the Vickrey–Clarke–Groves (VCG) auction have been shown to be strategy-proof for payoff-maximizing bidders (Clarke 1971, Groves 1973, Vickrey 1961). The payment rule in the VCG mechanism can be used to turn simple but also complex resource allocation problems into a strategy-proof and efficient economic mechanism, as long as the allocation problems can be solved to optimality. A number of authors have dealt with mechanisms with money in the domain of scheduling (Heydenreich et al. 2007).

Unfortunately, important scheduling problems cannot be addressed with auction mechanisms. First, monetary transfers are not allowed or desired. Second, cardinal utility and interpersonal utility comparison is too much to assume and we can only hope to get ordinal preferences. Finally, most scheduling problems are computationally hard and for realistic problem sizes an exact solution might not be tractable. However, the VCG mechanism requires exact solutions to the allocation problem.

Course assignment is a case in point. Students have preferences for different combinations of courses across the week. It is considered unacceptable to have students pay for course assignments in most environments, and we cannot expect from students the assignment of cardinal utilities to course bundles, only ordinal rankings of these bundles. Besides, the allocation problem is equivalent to a multi-unit combinatorial auction as we show in Section 2.1, which is known to be NP-hard (Lehmann et al. 2006). For larger problem sizes, such problems become intractable.

Currently, course schedules are often determined based on First-Come First-Served procedures (FCFS). As it is hard for students to influence whether they are first, second, or third in a row, this procedure is similar to random serial dictatorship (RSD), the only widely used and well-studied mechanism that is incentive-compatible and (Pareto) efficient. Students log on to the registration system at some time and select the best bundle of courses among the remaining course seats according to their preferences. Unfortunately, FCFS is not envy-free (Mas-Colell et al. 1995). Envy-freeness is a central notion of fairness and ensures that a student prefers the matching she is faced with to that offered to other students. The fact that FCFS leads to substantial envy in course scheduling leads students to switch assignments until late in the semester, and the resulting schedules for students are undesirable because tutorials are scattered across the week. This has been a growing concern at the Technical University of Munich (TUM) due to increasing student numbers over the past 10 years.

The theory of matching under preferences analyzes mechanisms which set incentives for participants to reveal their preferences truthfully without monetary transfers. Since winning the Noble Prize in Market Design in 2012, the field has drawn significant academic attention, with important applications in the matching of residents to hospitals, in school choice, or in kidney exchanges (Manlove 2013). Until recently, this theory was largely restricted to problems where participants have unit demand (e.g., a single course seat). However, new approaches have been developed which allow for more complex preferences such as preferences for packages of course seats or schedules and other complex constraints (Budish et al. 2013, Kamada and Kojima 2017, Nguyen et al. 2016). Interestingly, randomization and approximation turned out to be powerful tools for designing new mechanisms with good properties such as efficiency, envy-freeness, and incentives for truthful reporting in the presence of complex preferences (Nguyen et al. 2016). So far, such mechanisms have not been used in the field and the empirical performance of randomized mechanisms has not been studied. We analyze randomized approximation mechanisms in the context of course scheduling in the field and in the lab. Course assignment will be our focus in this study, but we discuss alternative applications at the end of this study.

1.1. Course Assignment and Distributed Scheduling

While some universities use matching mechanisms such as the deferred acceptance algorithm (Diebold et al. 2014, Gale and Shapley 1962) or course bidding (Krishna and Ünver 2008, Sönmez and Ünver 2010), in most cases scarce course seats are allocated via FCFS policies. Although many course assignment problems are similar to the widely studied school choice problems (Abdulkadiroğlu and Sönmez 2003, Ashlagi and Shi 2016) with students having private preferences for one out of many courses, often students are interested getting combinations of courses across the week, not just a seat in a single course. Assigning schedules of courses has been referred to as the *combinatorial assignment problem* (CAP) (Budish 2011).

The need to assign course schedules rather than courses individually became apparent in an application of matching with preferences at the TUM that we will discuss. The TUM uses the deferred acceptance algorithm for two-sided matching problems and RSD for one-sided matching problems in situations where only one out of a single course seat is required (i.e., unit demand). These algorithms are used to assign students to seminars or practical courses, and every semester about 1500 students are being centrally matched.

However, in many situations, students' preferences do not only concern a single course. For example, in the first three semesters of the study programs there are large courses with hundreds of students. These courses include a lecture and small tutor groups. Students need to attend one tutor group for three to four courses in each semester. These tutor groups should not overlap and they should be adjacent to each other such that students do not have a long commute for each of the tutor groups individually. For example, a student might want to have two tutorials in the morning and one after lunch on a particular day to reduce his commute time, and he would have a strong preference for this schedule over one where the tutorials are scattered across the week. In any case, students have timely preferences over course schedules that need to be considered, which makes it a combinatorial assignment problem.

While FCFS is widespread, the *Approximate Competitive Equilibrium from Equal Incomes* mechanism (A-CEEI) provides an alternative, which was recently introduced by Budish (2011). This mechanism was implemented in the Course Match system and used at the University of Pennsylvania (Budish et al. 2017). In this system, students report their complete preferences over schedules of courses and the mechanism assigns a budget of fake money to each student that she can use to purchase packages (or schedules) of courses. Then an optimization-based mechanism computes approximate competitive equilibrium prices, and the student is allocated her most preferred bundle given the preferences, budgets, and prices. It is well known that serial dictatorships are the only strategy-proof and efficient mechanisms for multi-unit and also combinatorial assignment problems (Ehlers and Klaus 2003, Pápai 2001). Therefore, A-CEEI has to relax design goals such as strategy-proofness and envy-freeness to approximate notions. Budish and Kessler (2021) summarize the results of laboratory experiments.

The work was breaking new ground, but the A-CEEI mechanism is also challenging. First, the problem of computing the allocation problem in A-CEEI is computationally very hard (PPAD-complete) and the algorithms proposed might not scale to larger problem sizes required in the field (Othman et al. 2016). The problems reported in Budish and Kessler (2021) already take 48 hours to solve on a compute cluster based on an elaborate implementation of the mechanism using various types of heuristics to speed up the solution time. There is no guarantee that somewhat larger problem sizes can still be solved. Second, it is not guaranteed that a price vector and course allocation exists that satisfies all capacity constraints, which leads to clearing errors. Third, students might not be able to rank-order an exponential set of bundles, which is a well-known problem (aka. missing bids problem) in the literature on combinatorial auctions (with money) (Bichler and Goeree 2017). The latter is a general problem in CAP not restricted to A-CEEI, which we also focus on in our paper.

Randomization can be a powerful tool in the design of algorithms, but also in the design of economic mechanisms. Nguyen et al. (2016) recently provided two randomized mechanisms for one-sided matching problems, one with cardinal and one with ordinal preferences for bundles of objects. The mechanism for ordinal preferences is a generalization of probabilistic serial (Bogomolnaia and Moulin 2001), called Bundled Probabilistic Serial (BPS). Nguyen et al. (2016) show that this randomized mechanism is ordinally efficient, envy-free, and weakly strategy-proof. These appealing properties come at the expense of feasibility, but the constraint violations are limited by the size of the packages. In course assignment problems, the size of the packages is typically small (e.g., packages with three to four tutor groups) compared to the capacity of the courses or tutor groups (around 30 seats or more). Even if capacities of a tutor group are violated by two or three seats, and this does not happen too often, tutors could cope with it by adding a few more seats. The mechanism does not have a need for prices or budgets, and only ordinal preferences are required from the participants. Computationally, the mechanism runs in polynomial time. Actually, even large instances can be solved in minutes, which makes the mechanism a practical alternative to FCFS that is easy to implement for university administrators with appropriate preference elicitation, as described in this study.

These properties make BPS a very attractive and fair (in the sense of envy-free) alternative to the widespread FCFS or RSD mechanisms, which is of importance to operations management way beyond the course allocation applications, which we focus on in this study. In Section 6, we discuss alternative applications in production and operations management.

1.2. Contributions

We provide results of a mixed-methods study including field and laboratory experiments, a survey, and simulations to analyze the BPS mechanism in the context of course assignment. The field experiments are based on a first implementation of this randomized scheduling mechanism in a large-scale course assignment application. The evaluation of the field data from two initial pilots (with 1438 and 1778 students, respectively) led to the permanent adoption of the new system, the first real-world implementation we know of.

Preference elicitation is a central challenge in mechanisms for the combinatorial assignment problem, which is why we discuss it in more detail. In contrast to preference elicitation in multi-attribute decision making (Stewart 1995), it is the combinatorial explosion of possible packages of interest that leads to challenges for the decision maker. This problem has not received much attention in the literature yet. We introduce a new approach to rank the exponential set of bundle preferences a priori based on a domain-specific scoring function and elicit parameters of this scoring function from the participants. The approach is able to capture the specifics of the students' preferences in our domain, as we show.

The empirical analysis of BPS is the focus of our analysis. We ran two large-scale field experiments and compare BPS to FCFS (i.e., a simulation as RSD).

We focus on this comparison because RSD is the de facto standard in all application domains for this type of scheduling problem.[1] We report the size of the matchings (i.e., the number of students matched), the average rank of the bundle that a student got, the probability of being matched, the rank profile (i.e., how many students got their first, second, or third ranked bundle), and the popularity of BPS among students compared to FCFS (i.e., whether students would prefer the BPS outcome to FCFS). These properties are of central importance for the choice of mechanisms and important to understand beyond the theoretical properties of the mechanism.

The results of our field experiment show that BPS actually dominates FCFS in all of the empirical metrics introduced earlier in our field experiments. While the differences in most criteria are small, envy-freeness turns out to be an important advantage of BPS. The level of envy that we find in FCFS is substantial in spite of the limited complementarities in student preferences, who are only interested in packages with at most four tutor groups.

These results are based on the assumption that participants submitted their preferences truthfully in the field experiments. In contrast to FCFS, BPS is only weakly strategy-proof and the preferences elicited might not be the true preferences of bidders. We organized a laboratory experiment and conducted a survey with 80 students and found that students actually hide their least preferred bundles. Fortunately, such manipulation has little impact on the overall efficiency of the outcome, as we found in simulations where we even truncated the reports to only the top 10 or top 30 preferences of each student. The mechanism is robust as long as the top preferences reported reflect the true preferences of the students. The long tail of the preference reports primarily serves the purpose not to be assigned a random (and possibly very low preference) bundle in situations where none of the top preferences can be assigned.

In summary, our study shows that BPS is at least as good as FCFS in all criteria, and it clearly dominates FCFS as participants do not have envy in an expected sense. This advantage has to be traded off with the added complexity of preference elicitation and the fact that participants might manipulate or make mistakes, a risk that is much reduced in FCFS.

Our work is positioned at the intersection of operations management and information systems (Kumar et al. 2018). It is in line with a stream of research in information systems on multi-unit and combinatorial auctions, where monetary transfers can be used to allocate goods efficiently (Adomavicius et al. 2012, Bapna et al. 2010, Goetzendorff et al. 2015). In operations management, there is substantial recent literature on scheduling (Diamant et al. 2018, Lee et al. 2018, Yu et al. 2020, Zhou et al. 2021). Typically, this research addresses mathematical models and algorithms to compute optimal schedules. In some applications preferences of participants play a role on top of hard scheduling constraints, but these preferences are assumed to be given (Lemay et al. 2017). Our paper contributes to this literature and discusses an economic mechanism that elicits ordinal preferences from the participants in a truthful manner.

2. Problem Definition and Design Desiderata

We now define the combinatorial assignment problem (CAP) in the context of course assignment applications, desirable properties, and randomized mechanisms.

2.1. Problem Definition

Assigning objects to agents with preferences but without money is a fundamental problem referred to as *assignment problem with preferences* or *one-sided matching with preferences*. We use the term assignment or matching interchangeably. In course assignment, students express ordinal preferences, which need to be considered in the assignment. A *one-sided one-to-many course assignment problem* consists of a finite set of n students (or agents) S and a finite set of m courses (or objects) C with the *maximum capacities*$q = (q_1, q_2, \ldots, q_m)$.

In the *combinatorial assignment problem* in the context of course allocation, every student $i \in S$ has a complete and transitive preference relation $\succeq_i \in \mathfrak{P}$ over all possible subsets (or bundles) $b \in B$ of packages. Courses in our application are actually tutor groups and each tutor group belongs to one of ℓ classes. Students in our application can only select bundles with at most one tutor group in each of these classes. For example, a student might select a bundle with a course seat in a tutor group for mathematics on Monday at 1 pm, and another tutor group in software engineering two hours later, but no additional tutor group in mathematics or software engineering in this bundle. As a result, the possible size of a bundle b is $size(b) \leq \ell \ll m$. The web interface takes care that students only submit valid bundles, which have at most one tutor group for each of the ℓ classes and a size less than or equal to ℓ. A preference profile $\succeq = (\succeq_1, \ldots, \succeq_n) \in \mathfrak{P}^{|S|}$ is an n-tuple of preference relations. For most of the paper we will assume strict preferences, but we discuss indifferences in the conclusions section of our paper. We will discuss different ways to elicit these preferences $\succeq_i$ from all participants.

We can model feasible assignments with linear constraints. Thereby, bundles are described with binary vectors $b \in \{0, 1\}^m$, where $b_j = 1$ if a seat in course j is included in bundle b. We define the size of b with $size(b) = \sum_{j=1}^{m} b_j$, the number of different courses included in the bundle. Let x_{ib} be a binary variable describing if bundle b is assigned to student i. The supply constraints make sure that the capacity of the courses are not exceeded, and the demand constraints determine that each student can win at most one bundle.

$$\sum_{i \in S,\, b \in B} x_{ib} b_j \le q_j \forall j \in C \qquad \text{(supply)}$$

$$\sum_{b \in B} x_{ib} \le 1 \forall i \in S \qquad \text{(demand)}$$

$$x_{ib} \in \{0, 1\} \forall i \in S, b \in B \qquad \text{(binary)}$$

A *deterministic combinatorial assignment* (deterministic matching) is a mapping $M \subseteq S \times B$ of students S and bundles B of courses C. $\mathfrak{M}$ describes the set of all deterministic matchings. A matching is *feasible* if it is a feasible integer solution to the 2 and 1 constraints. *Random combinatorial assignments* (random matchings) are related to fractional assignments with $0 \le x_{ib} \le 1$ and random assignment mechanisms can be used to fractionally allocate bundles of course seats to students. A *lottery L* is a probability distribution over feasible deterministic matchings. A lottery of bundles induces probability shares over these bundles that satisfy 2 and 1 constraints. However, a fractional solution respecting 2 and 1 does not need to have a lottery over deterministic assignments.

For assignment problems with single-unit demands (*size*(*b*) = 1), the Birkhoff–von Neumann theorem (Birkhoff 1946, Von Neumann 1953) says that every fractional allocation can be written as a unique probability distribution over feasible deterministic assignments. That is, any random assignment can be implemented as a lottery over feasible deterministic assignments, such that the expected outcome of this lottery equals the random assignment. One can describe a random assignment as a bistochastic matrix P, where p_{ic} is the probability that student i is assigned to course c. The Birkhoff–von Neumann theorem shows that such a bistochastic matrix can be decomposed into a convex combination of permutation matrices, which describe feasible deterministic assignments.

Unfortunately, the Birkhoff–von Neumann theorem fails when bundles b with *size*(*b*)>1 need to be assigned. However, any fractional solution respecting the 2 and 1 constraints can be implemented as a lottery over integral allocations that violate the 1 constraints only by at most $\ell - 1$ course seats (Nguyen et al. 2016). This allows implementing the fractional solution as a lottery and leads to a practical mechanism with economic properties that would be incompatible with a deterministic mechanism, namely *efficiency, envy-freeness, and strategy-proofness*. These design desiderata need to be refined for randomized mechanisms because they can only be guaranteed in an expected sense.

2.2. Design Desiderata of Randomized Mechanisms

Stochastic dominance (SD) is the key concept among all of these definitions, as it provides a natural way to compare random assignments. Let Δ describe the set of all possible random matchings. With p_i we refer to the assignment of student i in the random matching p, and denote with p_{ib} the probability that student i gets allocated bundle b. We will omit the subscript i when it is clear which student is meant. Given two random assignments p, $q \in \Delta$, student *i SD-prefers p* to *q* if for every bundle b, the probability that p yields a bundle at least as good as b is at least as large as the probability that q yields a bundle at least as good as b. More formally, a student $i \in S$ *SD-prefers* an assignment $p \in \Delta$ over $q \in \Delta$, $p \succeq_i^{SD} q$, if

$$\sum_{b' \succeq_i b} p_{ib'} \ge \sum_{b' \succeq_i b} q_{ib'},\ \forall b \in B.$$

In other words, a student i prefers the random assignment p to the random assignment q if p_i stochastically dominates q_i. Note that $\succeq^{SD}$ is not a complete relation. That is there might be assignments p and q, which are not comparable with this relation. First-order stochastic dominance (p_i dominates q_i if $p_{ib} \ge q_{ib}$ for all $b \in B$) holds for all increasing utility functions and implies second-order stochastic dominance, which is defined on increasing concave (risk-averse) utility functions. In other words, risk-averse expected-utility maximizers prefer a second-order stochastically dominant gamble to a dominated one (Müller and Stoyan 2002).

One desirable property of matchings is *(Pareto) efficiency* such that no student can be made better off without making any other student worse off. A *deterministic* matching M is *efficient* with respect to the students if there is no other feasible matching M' such that $M'(i) \succeq_i M(i)$ for all students $i \in S$ and $M'(i)_i M(i)$ for some $i \in S$. One can generalize this to random matchings and lotteries:

Definition 1. (*Efficiency*). *A random assignment p* $\in \Delta$ *is called*

1. *ex post efficient, if p can be implemented into a lottery over Pareto efficient deterministic assignments.*
2. *ordinally efficient, if there exists no random assignment q stochastically dominating p, that is,* $\nexists q \in \Delta : \forall i \in S : q \succeq_i^{SD} p$ *and* $\exists i \in S : q_i^{SD} p$.

Ordinal efficiency comes from the Pareto ordering induced by the stochastic dominance relations of individual students. It can be shown that ordinal efficiency implies *ex post* efficiency (Bogomolnaia and Moulin 2001).

Fairness is another important design goal. A basic notion of fairness for randomized assignments is the *equal treatment of equals*, that is, students with identical preferences receive identical (symmetric) random allocations. A stronger property is envy-freeness.

Definition 2. (*Envy-Freeness*). *A random assignment* $p \in \Delta$ *is called*

1. (*strongly*) *SD-envy-free, if* $\forall i,\, j \in S : p_i \succeq_i^{SD} p_j$.
2. *weakly SD-envy-free, if* $\nexists i,\, j \in S : p_{ji}^{SD} p_i$.

SD-envy-freeness implies equal treatment of equals and means that student *i* weakly *SD*-prefers the random matching she is faced with to the random assignment offered to any other student, that is, a student's allocation stochastically dominates the outcome of every other student. For weak *SD*-envy freeness, it is only demanded that no student's allocation is stochastically dominated by the allocation of another student.

An assignment mechanism is an algorithm, which computes a matching *M* for given preferences of students. More formally, a *deterministic assignment mechanism* is a function $\chi : P^{|S|} \rightarrow \mathfrak{M}$ that returns a feasible matching $M \in \mathfrak{M}$ of students to courses for every preference profile of the students.

A *randomized assignment mechanism* is a function $\psi : P^{|S|} \rightarrow \Delta$ that returns a random matching $p \in \Delta$. The mechanism $\psi\psi(\succcurlyeq) = p$ is *ordinally efficient* if it produces ordinally efficient allocations. We call $\psi\psi$ *ex post Pareto efficient*, if *p* can be decomposed as a convex combination of Pareto optimal matchings. $\psi\psi$ is *symmetric*, if for every pair of students *i* and *j* with $\succeq_i = \succeq_j$ also $p_i = p_j$. This means that students that have the same preference profile also have the same outcome in expectation. A randomized mechanism is *envy-free* if it always selects an envy-free matching.

An important property of a mechanism is *strategy-proofness*. This means that there is no incentive for any student not to submit her truthful preferences, no matter which preferences the other students report.

Definition 3. (*Strategy-Proofness*). *Let* $\succeq \in P^{|S|}$ *be the (true) preference profile. A deterministic assignment mechanism* χ *is strategy-proof if for every student* $i \in S$ *and* $\succeq_i' \in P$ *we have* $\chi_i(\succeq) \succeq_i \chi_i(\succeq_i', \succeq_{-i})$.

Thereby, $\succeq_{-i}$ denotes the preference profile of all agents $i' \in S \setminus \{i\}$. It has been shown that participants in strategy-proof mechanisms such as the Vickrey auction do not necessarily bid truthfully in practice. Due to this, there was a recent discussion about obvious strategy-proofness of extensive form games (Li 2017). Intuitively, a mechanism is obviously strategy-proof if the dominant strategy is very easy to understand. For randomized mechanisms, we need to adapt the definition of strategy-proofness.

Definition 4. (*SD-Strategy-Proofness*). *Let* $\psi : P^{|S|} \rightarrow \Delta$ *be a random assignment mechanism and* $\in P^{|S|}$ *the (true) preference profile.*

1. $\psi\psi$ *is called* (*strongly*) *SD-strategy-proof if for every student* $i \in S$ *with* $\succeq_i' \in P \psi(\succeq) \succeq_i^{SD} \psi(\succeq i', \succeq_{-i})$.
2. $\psi\psi$ *is called weakly SD-strategy-proof if there exists no* $\succeq_i' \in P$ *for some student* $i \in S$ *such that* $\psi(\succeq_i', \succeq_{-i}) \succeq_i^{SD} \psi(\succeq)$.

In other words, an ordinal mechanism is strategy-proof if for any agent, the allocation resulting from misreporting is weakly stochastically dominated by the allocation from truthful reporting, with respect to an agent's true preferences. Weak strategy-proofness means that there may not be any student *i*, who strictly prefers $\psi(\succeq_i', \succeq_{-i})$ over the truthful outcome, but there may be students *i* who neither prefer $\psi(\succeq_i', \succeq_{-i})$ nor $\psi\psi(\succcurlyeq)$. We will omit the prefix *SD* for brevity in the following.

3. Randomized Mechanisms

We now discuss different randomized mechanisms for the combinatorial allocation problem that satisfy design desiderata discussed in the previous section. Much is known about assignment problems with single-unit demand. RSD selects a permutation of the agents uniformly at random and then sequentially allows agents to pick their favorite course among the remaining ones. Gibbard (1977) showed that random dictatorship is the only anonymous and symmetric (i.e., equals are treated equally), strongly *SD*-strategy-proof, and ex post efficient assignment rule when preferences are strict. Pycia and Troyan (2018) proved that RSD is the unique mechanism that is obviously strategy-proof, efficient, and symmetric in mechanisms without monetary transfers. A version of RSD is explored in simulations in the context of course matching Li (2020).

However, RSD is not always ordinally efficient, only ex post efficient, which is somewhat weaker (Bogomolnaia and Moulin 2001). Zhou (1990) actually showed that no random mechanism for assigning objects to agents can satisfy strong notions of strategy-proofness, ordinal efficiency, and symmetry simultaneously with more than three objects and agents. So, we also cannot hope for these properties in combinatorial assignment problems. RSD can also be applied to the combinatorial assignment problem. The Bundled Random Serial Dictatorship (BRSD) orders the students randomly and assigns the most preferred bundle which is still available to each student in this order. Although the package preferences take some toll on the runtime, it is still very fast.

First-Come First-Served (FCFS) can be interpreted as a serial dictatorship. Students log in during a certain registration time and then reserve the most preferred bundle of courses that is still available. Although the arrival process is not uniform at random, students have little control over who arrives first, which can be seen as a reasonable approximation of BRSD. While there is a certain time when the registration starts, hundreds of students log in simultaneously to get course seats and it is almost random who arrives first. We simulate FCFS via BRSD and run the algorithm repeatedly to get estimates for performance metrics of FCFS in expectation.

Probabilistic Serial (PS) (Bogomolnaia and Moulin 2001) is a mechanism for the assignment problem with unit demand, which has drawn considerable attention in the literature. PS produces an envy-free assignment with respect to the reported unit-demand preferences, and it is ordinally efficient, but it is only weakly *SD*-strategy-proof. The algorithm is such that agents simultaneously "eat" fractions of individual objects leading to a fractional allocation. As an example, suppose two students both have the highest priority for a course with only a single seat. The agents have equal rights and a uniform eating speed of 1 unit per minute. After 0.5 minutes, each have one half of this course seat (a probability share) before they start eating their next preferred item from those that are still left. If a student would prefer a probability share of a different good rather than the one she is eating at a point in time, then that preferred good must have already been eaten away. Bundled Probabilistic Serial (BPS) by Nguyen et al. (2016) is a generalization of PS to the CAP also computing fractional solutions. The BPS mechanism is also ordinally efficient, envy-free, and weakly strategy-proof if preferences are strict.

Informally, in BPS all agents *eat* their most preferred bundle in the time interval [0, 1] simultaneously with the same speed as long as all included objects are available. As soon as one object is exhausted, every bundle containing this object is deleted and the agents continue eating the next available bundle in their preference list. The duration with which every bundle was eaten by an agent specifies the probability for assigning this bundle to this agent. Then, in the polynomial time *lottery algorithm*, one computes at most d+1 integral points, the convex hull of which includes or is arbitrarily close to the fractional solution x^* which results from BPS. The lottery algorithm then returns a lottery over these d+1 integral vectors, which is close to x^* in expectation. Finally, one of the deterministic outcomes is drawn from the lottery. Such a deterministic outcome over-allocates each course by at most ℓ−1 seats. This means the two constraints are fulfilled and only the one constraints are relaxed (Nguyen et al. 2016). Overall, the BPS mechanism is ordinally efficient, SD-envy-free, weakly SD-strategy-proof, and under preferences that demand at most k objects can be implemented so that it over-allocates each good by at most k−1 units (Nguyen et al. 2016).

4. Preference Elicitation

The BPS mechanism described in the previous section and its properties require that the designer has access to a strict preference relation $\succcurlyeq i$ for all students and bundles $b \in B$. Note that with q course seats in m courses, $|B| = (q+1)^m$. Naturally, preference elicitation becomes a central problem in the implementation of BPS in the field. In what follows, we first introduce the environment and the problem for students before we discuss different approaches to elicit their preferences.

4.1. Overall Process and Background

The Department of Computer Science at the TUM has been using stable matching mechanisms for the assignment of students (with unit-demand preferences) to individual course seats in seminars and laboratories since 2014. The system provides a web-based user interface and every semester almost 1500 students are matched to laboratory courses or seminars via the deferred acceptance algorithm for two-sided matching.

First, the students specify their preferences for a set of courses in the upcoming semester until a certain deadline. Then the students are assigned to courses via the deferred acceptance algorithm and informed about their assignment via e-mail. Since the resulting outcomes of the deferred acceptance algorithm are envy-free, attempts to change to another course are very limited. Only a few students drop out before the semester starts and overall the assignment procedure has proven very stable in practice. The software did not support combinatorial assignments until 2017. Therefore, the web-based software was extended with BPS, the lottery mechanism for decomposing fractional solutions, and BRSD, which were described in the last section.

The overall process for the combinatorial assignment followed that of the standard matching process (with unit demand) that was in use since 2014. A few numbers from the initial field experiments should provide a background and information about the scale of the application. During the summer term in 2017, 1439 students in their second semester of computer science and information systems participated in the matching and they could choose tutorial groups from several courses, including linear algebra, algorithms, software engineering, and operations research. A computer science student could have preferences for up to 5760(=10·24·24) bundles[2] and an information systems student could have preferences for up to 5184(=9·24·24) bundles.[3] A computer science student in the winter term could have even more than 700,000 different bundle preferences.[4] To rank order such a long list of preferences, students need decision support.

4.2. Automated Ranking of Packages

A naive approach to preference elicitation would be to let the students rank bundles on their own by choosing the time slots they want to have in their preference list. It would take a significant amount of time to rank even a small subset, and the students would only be able to rank order a few course bundles leading to a substantial missing bids problem. At most, one could expect students to rank their top 10 or 20 preferences manually. However, there is no guarantee that they actually receive one of their top ranked bundle. Without a ranking of all preferences, students face the risk of either getting a random assignment or no assignment at all. The problem how to elicit ordinal preferences for an exponential set of course bundles has not received much attention, and we are only aware of one related paper by Budish and Kessler (2021) that we will compare with in Section 4.3.

We propose a new approach to address the missing bids problem. This approach is based on a domain-specific scoring function. After discussions with students, it turned out that a few obvious aspects matter for students. The students' preferences for the tutorials mainly concern their commute and the possibility of having free, large contiguous blocks of time (e.g., a day or a half-day) that they can plan for other activities (e.g., a part-time job). When they register for tutorials, they do not know the name of the tutor, so only timely preferences matter. In larger cities, the time that students spend in commuting is significant and students want to minimize the commute time. Also, long waiting times between courses are perceived as a waste of time, as it is often hard for them to work productively in several one- or two-hour breaks. The minimal length of the break and recovery time differed, however, as well as the maximum number of tutorials a student wanted to have per day or the times of the day available on each day. After the interviews, it was clear that after eliciting only a few parameters such as the minimal length of the breaks, the acceptable times of each day, and the maximum number of tutorials per day, one could define a scoring function that ranks all possible packages automatically. Students could inspect the resulting ranking, revise the parameters of the scoring rule or individual ranking results.

Figure 1 shows screen shots of the overall process. First, students choose the lectures and tutorials they are interested in. The selected lectures will be considered in the bundle generation as constraints, that is, if a time slot of a tutorial for one lecture overlaps with the time of another selected lecture, then it will no longer be considered in order to allow students to participate in the lecture. In the second step, the student marks available time ranges in a *weekly schedule* for each day (see Figure A.4 in Appendix A of the online companion). The day is partitioned into weekdays and time blocks of 30 minutes from 8:00 AM to 8:30 PM. If a tutorial is selected, all time slots of this tutorial will be highlighted with a specific color. Thus, students can see when the tutorials and lectures of interest take place.

A student can set a minimal amount of time for a lunch break and a minimal amount of time between two events (default value is 15 minutes). We also allow students to provide weights {1, …, 5} for the different days. That is, the students can express preferences over the days. With this information from students, we parameterize a scoring function to rank order the different combinations of course bundles.

An algorithm first generates bundles that satisfy all constraints and then ranks them based on the scoring functions. Bundles are generated one by one for all students. Finding the bundles that do not violate constraints of the students (e.g., lectures to be attended) can be cast as a constraint satisfaction problem. After the feasible bundles are generated, the bundles are scored. The score considers

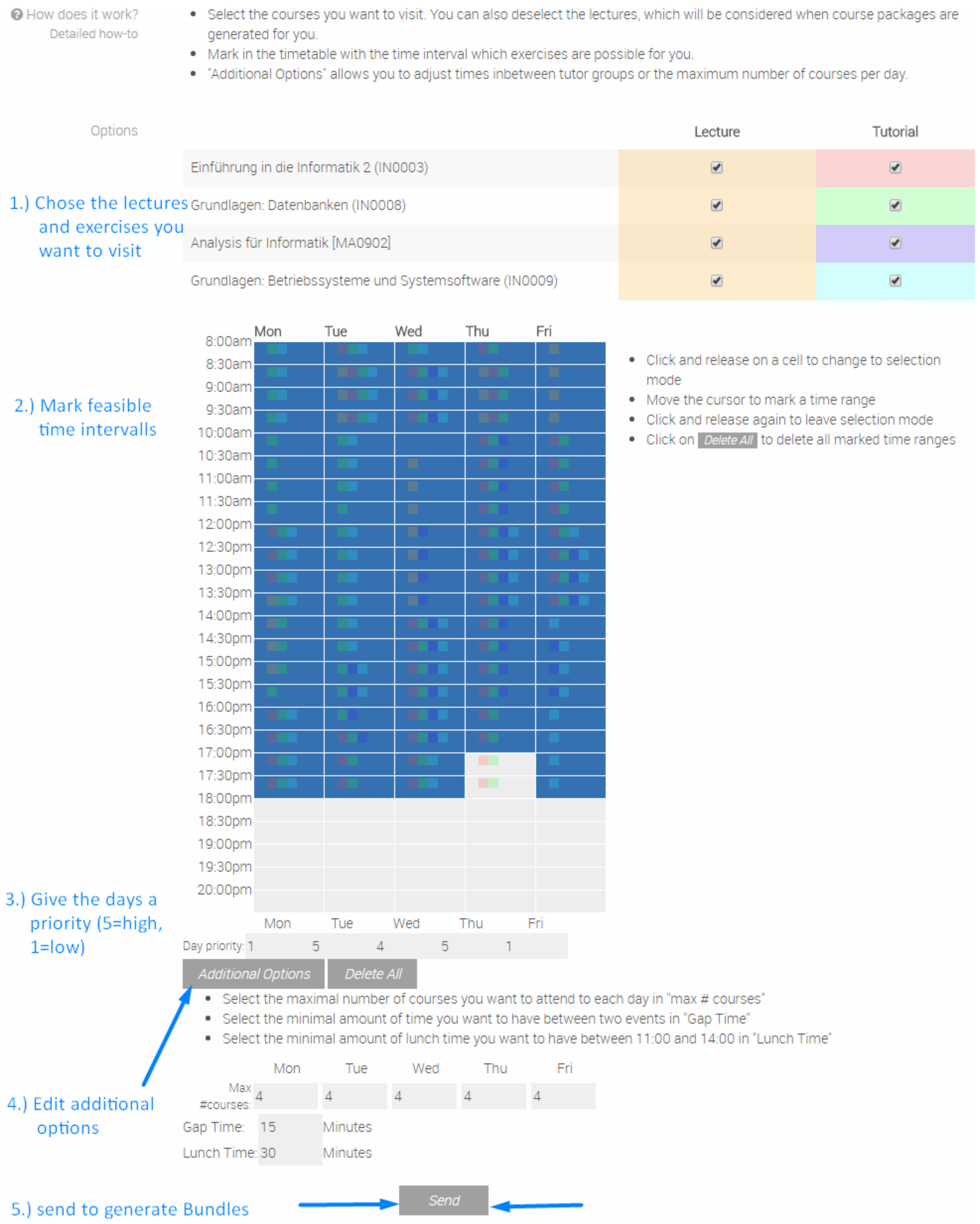
1. How many days a student needs to come to the university per week,
2. The preference ordering over the days,
3. The total time a student has to stay at the university each day, and
4. The length of the breaks between courses.

The score for a bundle b of courses across the week is the sum of the daily score ($score(b, day)$) for all weekdays d. The daily score is then computed as

$$score(b, day) = \left(\frac{w(b, day)}{sp(b, day)} \cdot f(sp(b, day) + br(b, day)\right) \cdot prio(day). \quad (4)$$

This score is scaled between 0 and 27.5 at a maximum and it considers how well the day is utilized with courses based on the preference parameters

Figure 1 Process to Rank-Order Packages [Color figure can be viewed at wileyonlinelibrary.com]



How does it work?
Detailed how-to

- Select the courses you want to visit. You can also deselect the lectures, which will be considered when course packages are generated for you.
- Mark in the timetable with the time interval which exercises are possible for you.
- "Additional Options" allows you to adjust times inbetween tutor groups or the maximum number of courses per day.

Options

1.) Chose the lectures and exercises you want to visit

	Lecture	Tutorial
Einführung in die Informatik 2 (IN0003)	✓	✓
Grundlagen: Datenbanken (IN0008)	✓	✓
Analysis für Informatik [MA0902]	✓	✓
Grundlagen: Betriebssysteme und Systemsoftware (IN0009)	✓	✓

2.) Mark feasible time intervalls

- Click and release on a cell to change to selection mode
- Move the cursor to mark a time range
- Click and release again to leave selection mode
- Click on Delete All to delete all marked time ranges

3.) Give the days a priority (5=high, 1=low)

	Mon	Tue	Wed	Thu	Fri
Day priority:	1	5	4	5	1

Additional Options Delete All

- Select the maximal number of courses you want to attend to each day in "max # courses"
- Select the minimal amount of time you want to have between two events in "Gap Time"
- Select the minimal amount of lunch time you want to have between 11:00 and 14:00 in "Lunch Time"

4.) Edit additional options

	Mon	Tue	Wed	Thu	Fri
Max #courses:	4	4	4	4	4

Gap Time: 15 Minutes
Lunch Time: 30 Minutes

5.) send to generate Bundles

Send

specified by the student. The time spent at the university per day *sp*(*b*, *day*) is considered relative to the time a student attends courses on that day (*w*(*b*, *day*)). These courses include tutorials and lectures. The ratio is used to weigh the score for a day (*f*(*sp*(*b*, *day*))). Hence, a day where students do not spend more time in breaks than the minimum number of minutes specified by the student maximizes the score. The function *f*(·) assigns points according to the number of hours spent at the university per day.

A second component in the daily score (*score*(*b*, *day*)) is the lunch break. Based on the student's preferences, for example, a 1-hour break would be considered best and therefore would lead to the highest score for the lunch break. Longer times or shorter times would lead to lower scores. The daily scores are then multiplied by the priority of the day [1..5]. If students do not have to visit the university at day *d*, they get a fixed score of 30 for this day. The overall score of a bundle *b* is the sum of the *score*(*b*, *d*) for all weekdays. As a result of this scoring rule, the more days the student can stay at home, the higher is the score of this bundle. As a simplified example, if a student had to come to the university on three different days to attend one course per day, this bundle would get a score of less than 25, but if he could attend all courses on a single day with minimal breaks, this bundle will get an overall score of more than 80 (for these three days). In other words, the scoring rule will place bundles that use a minimal number of days (ideally the most preferred days) with only a few breaks on top of the preference list. This would minimize the commute time and maximize the contiguous time a student can devote to learning or other activities. If the breaks between courses grow larger or courses take place on different or more days, this decreases the score. Ties are not impossible but almost never occur such that the algorithm typically generates a strict ranking of the feasible packages. Note that we only need an ordinal ranking of the course bundles at the end and the very score of a bundle does not matter. Also, the scores are not compared across students, only the ordinal ranking of the bundles.

Students enter their preferences into a database in parallel or sequentially during the preference elicitation phase and before the BPS algorithm is executed. After one of the students enters his preferences and generates a ranking of bundles, he might not be happy with the resulting rank order of bundles. In this case, he can manually rearrange the bundles in the ranking or he can go back and score again with different parameters (see Figure A.5 in the online companion). The first page shows the 30 top-ranked bundles. Note that ≈90% of the students received one of their top 10 ranked packages and only very few students received a package with a rank larger than 30. So, if a student manually inspects and confirms the ranking of the first 10–30 packages, this covers the most important quantile of the overall ranking list. This exercise can be done in a few minutes based on the generated ranking.

Of course, there are different ways of how such a scoring function can be specified. Obviously, any scoring function is only a heuristic helping students determine the strict ranking which is required by BPS, and any attempt to score and rank a large set of bundles can also be challenged (see discussion in Section 5.6). We have conducted extensive tests with students to understand how well the resulting ranking reflects their preferences. The feedback in these tests and that of the survey organized after the matchings was very positive, emphasizing the flexibility that it provides in specifying preferences.

The details of the scoring function are specific to our course assignment application. We argue, however, that one can find a reasonable scoring rule to help participants rank order in most application domains. For example, when assigning packages of time slots to carriers in retail logistics, these packages could simply be rank-ordered by computing the round-trip times for the various acceptable tours and rank by shortest time (Karaenke et al. 2019). Timely preferences, such as in our application, could also play a role in allocating surgeons to operating rooms, etc.

4.3. Challenges of Course-Level Scoring

Ranking an exponential set of packages is a general issue in course assignment problems, and one might ask for alternative methods. The only related work by Budish et al. (2017) that we are aware of will be discussed next. Budish et al. (2017) describes the preference elicitation used at the Wharton School of Business for the A-CEEI mechanism. Students could report cardinal item values on a scale of 1 to 100 for any course they were interested in taking. In addition, they could report adjustments for pairs of courses, which assigned an additional value to schedules that had both course sections together. Courses were then scored and transformed into an ordinal ranking over feasible schedules. The authors argue that they felt that "adding more ways to express non-additive preferences would make the language too complicated." Wharton also provided a decision support tool listing the 10 most-preferred bundles, which allowed students to inspect top-ranked schedules and modify the cardinal values. The authors write that "if the preference language given to students is not sufficiently rich ... then Course Match may not yield desirable results." Assigning course-level scores is fast and simple, but two problems make this method challenging to apply in our domain.

First, the ranking is sensitive to minor changes in the weights, which is a well-known issue in multi-criteria decision making with additive value functions. Evaluation is characterized by a substantial degree of random error, and the amount of error tends to increase as the decision maker attempts to consider an increasing number of attributes (or courses in our case) (Bowman 1963, Fischer 1972). Difficulties in the calibration of scores for each course can lead to substantial differences in the resulting ranking. It is very hard to precisely calibrate such weights in general.

Second, and more importantly, significant non-linearities arise due to the timely preferences of students in the assignment of tutorials in our course assignment application, making it impossible to describe the preferences via a course-level utility function as proposed by Budish et al. (2017). Even if three tutorials get a high number of points, this does not mean that their combination is preferable by a student as these tutorials might be on different days or have long breaks in between. To see this, we translated the ranking of packages into a set of inequalities with weights (w) as variables. Following revealed preference theory (Mas-Colell et al. 1995), we use these inequalities to understand whether there is any set of weights that would allow describing the ranking using a utility function $\sum_{i\in C} b_i w_i + \sum_{\substack{i,j\in C\\ j>i}} b_i b_j w_{ij}$. The function $r(b)$ describes the rank of a bundle, while b is again the binary parameter vector with each component $b_i \in \{0, 1\}$ showing whether a course $i \in C$ is part of a package or not. The objective function minimizes the sum of error variables ε in (REV). If there is any set of weights that could reflect the ranking of packages in our experiments without these error variables, the resulting optimal objective value would be zero. For every violation of a constraint one has to increase the respective error variable to a positive value.

We had preferences ranking 4000–12,000 bundles for the courses of the winter term. None of these settings could be solved with objective value zero, that is, the generated preference lists are not representable with a linear model with adjustment-terms used by Budish et al. (2017). Even if it was possible to find such a vector of course-level weights, it would probably be very difficult to parameterize by students. Overall, eliciting preferences for hundreds of bundles is a challenging problem and needs to be tailored to the application, but the quality of any mechanism for combinatorial allocation problems depends heavily on this input, which is the reason why we discussed this issue in more depth.

$$\begin{aligned}
&\underset{\text{s.t.}}{\text{Min}}\, err(\varepsilon) = \sum_{b\in B} \varepsilon_b + \sum_{\substack{i,j\in C\\ j>i}} \varepsilon_{ij} \quad \text{(REV)}\\
&\sum_{i\in C} b_i w_i + \sum_{\substack{i,j\in C\\ j>i}} b_i b_j w_{ij} + \varepsilon_b \ge \sum_{i\in C} b'_i w_i + \sum_{\substack{i,j\in C\\ j>i}} b'_i b'_j w_{ij} \forall b,\, b' : r(b') = r(b)+1\\
&w_i + w_j + w_{ij} + \varepsilon_{ij} \ge 0 \quad \forall i,\, j\in C,\, j>i\\
&w_i \in [0,\, 1] \quad \forall i \in C\\
&w_{ij} \ge -2 \quad \forall i,\, j\in C,\, j>i\\
&\varepsilon_b,\, \varepsilon_{ij} \ge 0 \quad \forall i,\, j\in C,\, j>i,\, b\in B.
\end{aligned}$$

5. Empirical Evaluation

We now provide the results of our empirical mixed-methods study. First, we discuss the empirical metrics used. Second, we provide results from our field experiments. We take the preferences elicited for BPS in the field, compute another matching with BRSD with these preferences, and compare the various metrics for BPS and BRSD. As indicated in Section 3, BRSD is used as a proxy for the results of FCFS. Although BPS is weakly SD-strategy-proof, we do not know from the field experiment whether students indeed report their true preferences. Therefore, we organized a laboratory experiment with induced preferences. This laboratory experiment was complemented by a survey where we explicitly asked students how well they were able to specify their preferences and whether they were intentionally manipulating their induced preferences or not. The results of both the laboratory experiments and the survey indicate that students truncate their least preferred packages, that is, the ones including days for which they have a very low preference. In Subsection 5.5, we report results of simulations, where we truncate the original preferences to the top 10, 20, or 30 only, in order to understand how robust the efficiency results are against manipulation or imprecisions in the preferences of higher ranks.

5.1. Metrics

In this subsection, we introduce empirical metrics that allow us to compare BPS and BRSD in our experiments. The size and the average or median rank can be used as efficiency metrics. The *size* of a matching simply describes the number of matched agents. The *average rank* is only meaningful in combination with the size of the matching because a smaller matching could easily have a smaller average rank. We report the average rank because it has been used as a metric to gauge the difference in welfare of matching algorithms in Budish et al. (2017) and Abdulkadiroğlu et al. (2009), two of the few experimental papers on matching mechanisms.

The *profile* contains more information as it compares how many students were (fractionally) assigned to their first choice, how many to their second choice, and so on. The profile of two matchings is not straightforward to compare. We wanted to compare multiple profiles based on a single metric, and decided to use a metric similar to the *area under the curve of a receiver operating characteristic* in signal processing (Hanley and McNeil 1982) which was also used in Diebold and Bichler (2017). The *area under the profile curve ratio* (AUPCR) is the ratio of the area under the profile curve (AUPC) and the total area (TA) and is scaled between 0 and 100%, where the AUPC describes the integral below the profile curve. The AUPCR up to a specific rank n is equal to the probability that a matching mechanism will match a randomly chosen student higher than his n-th preference.

Definition 5. (*AUPCR*). *Let C be the possible courses with $c \in C$ and Q be the sum of all capacities, regarding the students $i \in S$ the AUPCR is defined as follows:*

$$\begin{aligned} TA(M) &= |C| \cdot min\{|S|, Q\} \\ AUPC(M) &= \sum_{r=1}^{|C|} |\{(i, c) \in M | rank(i, c) \leq r\}| \\ AUPCR(M) &= \frac{AUPC(M)}{TA(M)}. \end{aligned}$$

Aside from efficiency, fairness, and strategy-proofness, *popularity* was raised as a design goal. An assignment is called popular if there is no other assignment that is preferred by a majority of the agents. Popular deterministic assignments might not always exist, but popular random assignments exist and can be computed in polynomial time (Kavitha et al. 2011). Unfortunately, Brandt et al. (2017) have proven that popularity is incompatible with very weak notions of strategy-proofness and envy-freeness; however, it is interesting to understand the popularity of BPS vs. BRSD. In our empirical evaluation, we analyze whether BPS or FCFS are more popular. To measure popularity, we first define the function $\phi_i(b, b') : B \times B \to \{\pm 1, 0\}$ associated with the preference relations:

$$\phi_i(b, b') = \begin{cases} +1 & \text{if} b_i b' \\ -1 & \text{if} b'_i b \\ 0 & \text{else.} \end{cases}$$

Definition 6. (*Popularity*). *A random assignment $p \in \Delta$ is more popular than an assignment q, denoted $p \blacktriangle q$, if pop $(p, q) > 0$ with*

$$pop(p, q) = \sum_{i \in S b,} \sum_{b' \in B} p_{ib} \cdot q_{ib'} \cdot \phi_i(b, b').$$

A random assignment p is *popular*, if $\nexists q \in \Delta$: $q \blacktriangle p$.

5.2. Field Experiments

The overall process of the field experiment and how students submitted preferences was explained in the Section 4. Table 1 provides the main evaluation metrics, not only for the initial two field experiments in the summer term 2017 and the winter term 2017/18, but also those for the subsequent two years in which the combinatorial assignment was carried out. Taking the preferences for bundles of tutor groups elicited for the BPS allows for a comparison with BRSD on equal footing. To really compare the result of BPS and BRSD, we actually would have to run the BRSD for all permutations of the students. A single run is clearly not sufficient, as we need to compare probability distributions. Note that computing probabilities of alternatives in RSD explicitly is a computationally very challenging (#P-complete) problem (Aziz et al. 2013). Therefore, we ran BRSD 1000 to 1,000,000 times with the same preferences but random permutations of the order of students and derived estimates for the different metrics. Since these results are very close, one can assume that one million runs of BRSD generate a good approximation to the (real) induced random matching.[5] Details on the BPS lottery of the summer-term instances can be found in the online companion in Appendix B. We discuss these metrics in more detail.

The computation times were negligible for BRSD (0.007 seconds per run). BPS required 0.12 seconds computation time with an additional 6 minutes for the lottery algorithm in the summer term 17, for example. This shows that BPS is a practical technique even for large assignment problems, which can be run on standard computers. In contrast, solving deterministic scheduling problems of this size as integer problems would typically be intractable. Also, the A-CEEI mechanism and the Course Match system that is used for related applications is computationally very demanding (Budish et al. 2017).

In terms of average rank, average size, and the probability of being matched to one of the first 100 ranks, BPS is better than BRSD in all four instances, but not by much. The AUPCR is sometimes slightly higher for BPS, sometimes for BRSD. In addition, Table 2 reports the probability of being matched to one of the top 10 ranks and the AUPC in percentage for BPS and BRSD. Differences in these metrics are minimal.

Our field experiments confirm the theoretical result that BPS is (strongly) *envy-free* (see Definition 2) as can be seen by the respective entries in Table 1. In contrast, BRSD is neither weakly nor strongly envy-free. In the summer term 17, 1067 students do not fulfill the envy-freeness condition (see Definition 2), from which 374

Table 1 Summary Statistics for all Instances

	Summer term 17		Winter term 17/18		Winter term 18/19		Winter term 19/20	
Metric	BPS	BRSD	BPS	BRSD	BPS	BRSD	BPS	BRSD
Exp. rank	2.202	2.208	1.97	1.98	2.55	2.57	1.5670	1.5697
Exp. size	1086.7	1085.8	1602.9	1600.9	1549.1	1548.7	1414.90	1413.5
Prob. match(%,top 100)	76.79	76.73	92.33	92.14	92.04	92.02	94.39	94.30
AUPCR (%)	74.74	75.08	88.94	88.81	88.06	89.56	92.38	92.28
Weak envy	0	374	0	454	0	482	0	617
Strong envy	0	1067	0	1200	0	1262	0	1439
Popularity	2.97		3.09		1.80		2.35	
SD-preference	(669|94)		(740|125)		(772|92)		(1019|35)	

Table 2 Rank Profiles for all Instances

Rank	1	2	3	4	5	6	7	8	9	10
Summer Term 2017 – BPS										
Prob match(%)	54.18	5.69	4.54	2.02	1.51	0.94	1.17	0.94	1.14	0.61
AUPC in (%)	54.18	59.87	64.41	66.44	67.94	68.88	70.04	70.98	72.13	72.74
Summer Term 2017 – BRSD										
Prob match(%)	53.97	5.73	4.54	2.05	1.53	0.93	1.18	0.95	1.15	0.61
AUPC in (%)	53.97	59.70	64.24	66.29	67.82	68.75	69.93	70.88	72.03	72.64
Winter Term 17/18 – BPS										
Prob match(%)	73.58	7.07	3.40	1.66	1.04	0.70	0.47	0.45	0.37	0.30
AUPC in (%)	73.58	80.66	84.05	85.71	86.76	87.46	87.92	88.37	88.74	89.04
Winter Term 17/18 – BRSD										
Prob match(%)	73.45	7.05	3.38	1.67	1.04	0.70	0.49	0.44	0.36	0.31
AUPC in (%)	73.45	80.50	83.88	85.55	86.59	87.30	87.78	88.22	88.58	88.89
Winter Term 18/19 – BPS										
Prob match(%)	67.84	7.94	4.24	2.28	1.77	0.82	0.76	0.71	0.40	0.55
AUPC in (%)	67.84	75.78	80.01	82.29	84.06	84.87	85.63	86.33	86.73	87.27
Winter Term 18/19 – BRSD										
Prob match(%)	67.73	7.97	4.25	2.28	1.77	0.82	0.76	0.69	0.40	0.55
AUPC in (%)	67.73	75.70	79.95	82.23	84.00	84.82	85.58	86.27	86.67	87.22
Winter Term 19/20 – BPS										
Prob match(%)	75.33	8.87	3.66	1.74	1.17	0.73	0.49	0.45	0.22	0.29
AUPC in (%)	75.33	84.20	87.86	89.60	90.77	91.50	91.99	92.43	62.65	92.94
Winter Term 19/20 – BRSD										
Prob match(%)	75.19	8.88	3.67	1.76	1.18	0.73	0.49	0.45	0.22	0.29
AUPC in (%)	75.19	84.07	87.74	89.50	90.67	91.40	91.89	92.33	92.55	92.84

students do not even fulfill the weak envy-freeness condition (see BRSD in Table 1). Similarly, for the winter term 1200, students do not *SD*-prefer their outcome over the outcomes of every other student, and 454 of those students even prefer an outcome of another student. We see the same pattern in later years.

A positive popularity score as described in Definition 1 means that BPS is more popular than the BRSD outcome and the score for BPS is 2.97 for the summer term 17 and 3.09 for the winter term 17/18, for example. In all four instances, BPS is more *popular* than BRSD. For the data from the winter term 17/18 740, students prefer BPS to BRSD, while 125 students prefer BRSD to BPS. 871 students are indifferent. The syntax for the *SD-preference* is the number of students preferring (BPS|BRSD). It shows that BPS is preferable to BRSD according to *SD*-preference.

5.3. Laboratory Experiments

In order to study whether students submit their preferences truthfully, we report results of a laboratory experiment. The experiment was conducted from June 8, 2020 until June 16, 2020 at the TUM. The participants were able to participate in the experiment online within a period of nine days. In total, 91 students participated. Eleven students either did not complete the matching or the survey, which is why they were excluded, resulting in 80 participants that were evaluated. The students were all studying computer science or information systems, which is representative for the type of students that usually participate.

After having sent out the link to the experiment platform with an individualized authorization code, participants could start the experiment by logging into the experiment's platform. The platform we used for this experiment was provided by soscisurvey.6 We ensured an anonymous treatment of participants' data at all times. Furthermore, the authorization code, allocated by the platform, allowed for an anonymous processing as well as stopping and continuation of a student's progress in the experiment if necessary. At the beginning, participants could download instructions, which explained the experimental setting, the procedure, and rules of the experiment (see online companion D). On top of that, watching an additional instruction video was mandatory for being able to continue the experiment. In the video, we gave an overview of the instructions and explained the functionality of the matching tool.

Before students could start working with the matching tool, they had to take a quiz so that we could make sure that they understood the environment and the task. After they participated, they had to fill in a survey, which we discuss in the next subsection. Budish and Kessler (2021) argue that, at least in the setting of preferences over course topics, no monetary incentive is needed because most participants will take decisions as seriously as in a real setting. However, monetary incentives increase the external validity of the experiment, which is why we introduced a winning lottery. The reward mechanism was such that, the better the outcome of the matching fit the induced ranking of bundles, the higher the participant's chances of winning in a lottery of five times 40 EUR. This mitigated the risk of participants submitting random input.

The experimental design was simple. The only treatment in our experiments was the variation in preference profiles (see Figure 2). These were modeled after preference profiles we found in field data. In the analysis, we were only interested in the degree to which students reported their induced preferences truthfully.

The results of the experiment showed a clear pattern. Students often omitted the least preferred day in her induced preferences when reporting, as can be seen in Table 3. By least preferred day, we define the day or days in the preference profile with the lowest preference weight.

Aside from omitting a day altogether, students could also reduce the number of hours acceptable on a day. We state that a day is de-emphasized by choosing a time interval of less or equal than 2 hours, although a student would be available all day. Table 4 shows those days that were either omitted or de-emphasized. These simple descriptive statistics show a very clear profile that students truncate their reports and do not report low preferences. These findings were supported by a survey we took among the students participating in the laboratory experiment.

5.4. Survey Results

The survey was taken right after students submitted their preferences but before they learned the results such that the results do not influence the survey results in Table 5. The responses indicate that the majority of the students responding found the system easy to use and that they could express their preferences well. While 76.2% of the participants had no difficulties selecting time intervals in the weekly schedule (Question 1), 59.9% found it easy to rearrange the ten pre-ranked bundles (Question 2). Also, 69.9% of participants found their given preferences to be realistic (Question 6). This is in accordance to the finding that 73.7% of the respondents could infer their preferred tutorials from their given preferences (Question 7). A high proportion of participants (86.2%) could express their day priorities with ease (Question 9).

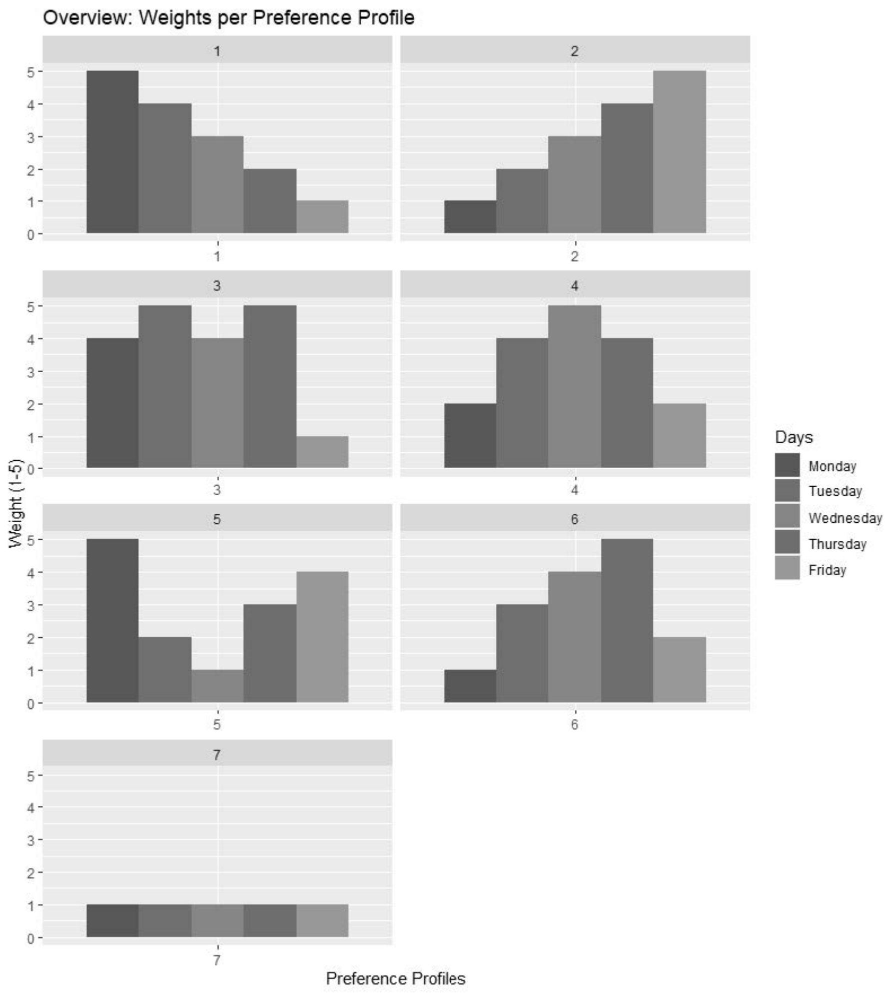
As the given preferences were clearly visible on the same scale as the day priorities, 91.2% of the participants indicated that they expressed their preferences in the matching tool in accordance with the given day priorities (Question 10). We furthermore see that 98.7% of participants perceive their preference reporting as being truthful (Question 11). However, 27.5% of the students report that they were hiding the least preferred time slots (Question 13). According to students' self-reporting, manipulation is mainly done by de-emphasizing least preferred days in the input of time slots (Question 14).

5.5. Sensitivity Analysis with Truncated Preferences

The fact that a part of the students indicate that they did not report low preferences truthfully is concerning. In FCFS, students only provide their single best package at the point in time when they log in. This is simple, intuitive, and obviously strategy-proof. This property has to be traded off against the level of envy in FCFS. So, the question is how truncation of the reports (omitting the least preferred bundles) impacts the overall results. We ran simulations where we took into account only the top 10, 20, or 30 bundles in the preference relations of all participants. This can be considered an extreme case of truncation. In our notation, BPS-10 reports the statistics for those instances, taking into account only the top 10 preferences. The resulting summary statistics can be found in Tables 6 and 7. Table C.8 in Appendix C in the online companion shows the rank profiles with and without truncation for all instances.

Interestingly, the impact of truncation is low. This is due to the fact that most students could be matched to one of their top 30 preferences. If we take into account only the top 10 preferences, the

Figure 2 Perference Profiles with Weights for Individual Days of the Week. A Higher Weight Means a Higher Preference



expected rank improves at the expense of the number of people matched. In other words, if a student is not assigned at all, this leads to less competition for the highly ranked bundles and to higher average ranks overall. With respect to popularity, more students prefer the results with truncated valuations. Except from WS19/20 BPS-30, students SD-prefer the version with truncation of the preferences to that with all preferences taken into account.

5.6. Discussion

In theory, the comparison between BRSD (as a proxy for FCFS) and BPS is clear: BRSD is obviously strategy-proof, efficient, but not envy-free. In contrast, BPS is weakly SD-strategy-proof, SD-envy-free, and ordinally efficient. An empirical analysis allows us to consider various empirical metrics to measure the quality of the matching beyond efficiency. Also, lab experiments help us understanding whether participants are indeed truthful in a mechanism that is weakly SD-strategy-proof.

Overall, we find that BPS dominates BRSD on most empirical metrics in our empirical evaluation (such as rank, size, and probability of being matched), but that the differences are very small. For the profile curves (AUPCR) there is no clear winner, which is interesting given the fact that only a small number of preferences per student are considered via BRSD.

There are a number of reasons that help in explaining the close performance of BPS and BRSD in these metrics. First, Che and Kojima (2010) have found that RSD and probabilistic serial become equivalent when the market becomes large, that is, the random assignments in these mechanisms converge as the number of copies of each object type grows and the inefficiency of RSD becomes small. Our empirical results suggest that differences might also be small in large combinatorial assignment markets with limited complementarities.

Second, ordinal preferences do not allow expressing the intensity of preferences. Suppose there are two students who both prefer course c_1 to c_2, each having one course seat only. No matter who gets

Table 3 Number of Omitted Days Per Preference Profile in Total. Boxed Entries Mark the Least Preferred Day in the Respective Profile. The Right Column Denotes the Number of Students who were Assigned this Profile

Preference profile	Monday	Tuesday	Wednesday	Thursday	Friday	Participants
1	0	0	0	1	3	12
2	4	0	0	0	0	12
3	1	0	0	0	4	13
4	1	0	0	0	1	12
5	0	0	5	0	0	10
6	5	0	0	0	1	11
7	0	0	1	0	0	10

Table 4 Number of de-emphasized and omitted days per preference profile in total. Boxed entries mark the least preferred day in the respective profile. The right column denotes the number of students who got this profile assigned

Preference Profile	Monday	Tuesday	Wednesday	Thursday	Friday	Participants
1	0	0	0	1	6	12
2	7	6	0	0	0	12
3	1	0	0	0	11	13
4	7	2	0	0	6	12
5	0	6	8	0	0	10
6	9	4	0	0	4	11
7	1	3	2	0	0	10

Table 5 Survey Results, Values in % for "Strongly Disagree" (1) to "Strongly Agree" (5)

		Strongly disagree				Strongly agree	Don't know
Question		1	2	3	4	5	
1	I had no problem in selecting my time intervals in the weekly schedule.	2.5	10.0	11.2	31.2	45.0	0.0
2	The ranking of the generated bundles was easy.	1.2	12.5	25.0	31.2	28.7	1.2
3	The instructions on the matching system were sufficient.	2.5	2.5	11.2	32.5	51.2	0.0
4	I found the matching tool too complicated to understand.	43.7	41.2	8.7	6.2	0.0	0.0
5	I already had experience in using the matching tool before the experiment.	3.7	3.7	10.0	82.5	0.0	0.0
6	I felt like my day priorities were realistic.	1.2	15.0	13.7	36.2	33.7	0.0
7	I could easily infer my preferred tutorials from my day priorities.	1.2	7.5	15.0	38.7	35.0	2.5
8	The generated sets (bundles) of tutorials met my expectations.	2.5	17.5	8.7	36.2	35.0	0.0
9	I was able to express my day priorities well in the matching tool.	1.2	3.7	8.7	30.0	56.2	0.0
10	I was expressing my preferences in accordance with my day priorities.	1.2	1.2	6.2	15.0	76.2	0.0
11	I indicated my preferences truthfully.	0.0	0.0	1.2	15.0	83.7	0.0
12	I was strategically hiding some of my *most* preferred time slots.	75.0	18.7	3.7	1.2	1.2	0.0
13	I was strategically hiding some of my *least* preferred time slots.	47.5	13.7	11.2	15.0	12.5	0.0
14	I was not selecting time slots for days which had the lowest priority in my day priorities.	22.5	21.2	7.5	10.0	33.7	5.0

course c_1, the average rank and size of the matching as well as the profile will be the same even though one student might desperately want to attend c_1, while the second student only has a mild preference for c_1. Without cardinal information about the intensity of a preference the differences in aggregate metrics can be small.

Third, an earlier comparison of FCFS with a deferred acceptance algorithm by Diebold et al. (2014) in environments with unit demand for a single course seat also showed that FCFS yielded surprisingly good results. While the average rank of FCFS was worse, the size of the matching resulting from FCFS was significantly larger compared to that from the deferred acceptance algorithm. For the combinatorial assignment problem, BPS also had a slightly larger average size. It is important to understand these trade-offs for applications of matching in practice.

A central theme of this study is that care needs to be taken in how preferences are elicited in any form of combinatorial assignment. Although our sensitivity analysis shows that the top preferences matter most, oversimplified scoring functions might not reflect the real preferences of participants and lead to bad outcomes for individuals. Our laboratory experiments provide empirical evidence that students attempt to

Table 6 Summary Statistics for all Truncated Instances SS17 and WS 17/18

	Summer term 17				Winter term 17/18			
	BPS	BPS-10	BPS-20	BPS-30	BPS	BPS-10	BPS-20	BPS-30
Exp. rank	2.20	1.96	2.13	2.14	1.97	1.51	1.76	1.90
Exp. size	1086.7	1077.8	1085.6	1085.8	1602.9	1581.8	1595.2	1600.3
Prob. match(%,top 100)	76.79	76.17	76.72	76.74	92.33	91.17	91.89	92.19
AUPCR (%)	74.74	74.34	74.66	74.65	88.94	89.18	89.17	89.02
Popularity		0.52	−0.26	0.18		−8.49	−3.11	−0.68
SD-preference		(76|737)	(199|489)	(177|468)		(31|1258)	(53|1126)	(295|660)

Table 7 Summary Statistics for all Truncated Instances WS 18/19 and WS 19/20

	Winter term 18/19				Winter term 19/20			
Metric	BPS	BPS-10	BPS-20	BPS-30	BPS	BPS-10	BPS-20	BPS-30
Exp. rank	2.55	1.58	1.98	2.19	1.57	1.41	1.52	1.57
Exp. size	1549.1	1513.6	1530.9	1537.8	1414.9	1406.2	1412.4	1414.8
Prob. match(%,top 100)	92.04	89.93	90.96	91.37	94.39	93.81	94.22	94.39
AUPCR (%)	88.06	88.27	88.34	88.26	92.38	91.74	92.36	92.38
Popularity		−10.45	−2.22	−1.07		−1.28	0.11	0.24
SD-preference		(45|1091)	(102|919)	(141|861)		(86|701)	(300|396)	(455|271)

provide their preferences truthfully. However, they might make mistakes and they frequently truncate their least preferred bundles. In contrast, preference elicitation for FCFS is as simple as possible.

6. Conclusions and Managerial Implications

Many organizations have distributed scheduling problems where ordinal and private preferences of participants matter. Designing incentive-compatible, efficient, and envy-free mechanisms for such problems without monetary transfers appeared impossible until recently. RSD is the de-facto standard for such types of scheduling problems, but it is not envy-free, which often leads to concerns. Bundled Probabilistic Serial uses approximation and randomization and satisfies randomized notions of efficiency, strategy-proofness, and envy-freeness.

We have studied randomized approximation mechanisms in the context of course assignments, which allowed us to run large-scaled field experiments. The informational requirements of BPS about student preferences are challenging, because a strict rank-ordering of exponentially many bundles is needed. As in many scheduling applications, student preferences in our course assignment application are about times of the week. We introduced a way to rank order the many possible schedules based on a few parameters. The feedback of students was that this automated ranking met their preferences well and we argue that this is a good way to address the missing bids problem in similar applications. Of course, other domains might have different requirements and the way how preferences are elicited needs to reflect the domain specifics. We argue that for most domains it will be possible to define a good scoring function that helps participants finding an initial ranking. Together with the possibility to manually rearrange the top ranked bundles, this is a practically viable approach to address the preference elicitation problem in the combinatorial assignment problem with time-dependent and ordinal preferences.

Although BPS provides a powerful new alternative to scheduling problems with private preferences, there are trade-offs. First, in contrast to FCFS, the BPS mechanism is not obviously strategy-proof and a part of the students in the survey already indicated that they hid their least preferred time slots strategically. This might partly be due to the fact that students were unexperienced with this new mechanism. Second, the assumption of strict preferences is strong in the presence of exponentially many bundles. Unfortunately, extending PS or BPS to preferences with ties is not without loss. On the one hand, Katta and Sethuraman (2006) extended PS to preferences with indifferences and showed that it is not possible for any mechanism to find an envy-free, ordinally efficient assignment that satisfies even weak strategy-proofness, as in the strict preference domain. On the other hand, with indifferences and random tie-breaking efficiency cannot be guaranteed. Our preference elicitation technique generated a strict and complete ranking of course bundles based on a few input parameters and is one way to address these issues.

Implementing and testing new artifacts in organizations is challenging and we are grateful for the opportunity to run a large-scale field experiment at the TUM. Such pilots are very valuable but also hard to organize in most organizations. This is particularly true for a non-standard mechanism such as BPS, which involves optimization and randomization. We report the results of field experiments with Bundled Probabilistic Serial for course scheduling, and show that it performs well on a number of criteria including average rank, average size, and the probability of a matching. The matching based on BPS is also more popular than the one resulting from FCFS on average based on the preferences submitted for BPS.

Most importantly, however, the level of envy in FCFS is significant, even though the size of the packages that can be submitted is limited to the number of classes (three to four groups per package) in our course assignment application. This has actually always led to complaints from students in the past. Many students wanted to change course seats with peers, which led to administrative overhead and complaints about bad course schedules for students. These complaints vanished after the BPS mechanism was adopted. The BPS mechanism is now an accepted tool among students to deal with the assignment of multiple tutor groups, and its introduction is considered a success as expressed by a letter from the Head of Academic Programs.

The study has implications for the broader field of production and operations management because many other applications share similar characteristics: participants have private and ordinal preferences for packages of objects, and monetary transfers are either not allowed or not desired. For example, in a recent paper an auction mechanism was used for the allocation of time slots at warehouses to carriers in retail logistics (Karaenke et al. 2019). Congestion at loading ramps is a significant problem in practice and it deteriorates the efficiency in retail logistics. It is well known that this is due to a lack of coordination among carriers, who all plan their routes independently. BPS could provide an alternative to FCFS that is envy-free and does not require monetary transfers. The problem is related to the literature on scheduling truck arrivals (Ou et al. 2010), and more generally to timetabling or conference scheduling (Sampson 2004). Similarly, various types of workforce scheduling share similar characteristics. This includes the assignment of nurses or doctors to shifts or crews to flights of airlines. In summary, if envy-freeness matters, the elegant BPS mechanism has a number of attractive properties, which otherwise suffer from computational hardness of the allocation problem and strategic manipulation. The results of this study provide useful insights to other organizations that want to adopt distributed scheduling mechanisms in lieu of simple FCFS heuristics.

Acknowledgments

We are grateful for valuable feedback from Nguyen Thanh and Rakesh Vohra. Financial support is acknowledged from the German Research Foundation (DFG) [Grant BI-1057 9/1].

Notes

[1]A-CEEI and the Course Match implementation provide an intricate solution with years of development effort and various heuristics used in the backend. Due to the computational cost of the mechanism and the implementation challenges, in our comparison we focus on the widely used RSD mechanism.
[2]Consisting of tutorials for the courses: linear algebra, algorithms, software engineering.
[3]Consisting of tutorials for the courses: operations research, algorithms, software engineering.
[4]The computer science students needed tutorials from all four classes (<22·25·26·52).
[5]For brevity, we omit the number of runs in the following and only report the results for one million runs of BRSD.
[6]https://www.soscisurvey.de/

References

Abdulkadiroğlu, A., T. Sönmez. 2003. School choice: A mechanism design approach. *Am. Econ. Rev.* **93**(3): 729–747.

Abdulkadiroğlu, A., P. Pathak, A. Roth. 2009. Strategy-proofness versus efficiency in matching with indifferences: Redesigning the NYC high school match. *Am. Econ. Rev.* **88**(5): 1954–1978.

Adomavicius, G., S. Curley, A. Gupta, P. Sanyal. 2012. Effect of information feedback on bidder behavior in continuous combinatorial auctions. *Management Sci.* **58**(4): 811–830.

Ashlagi, I., P. Shi. 2016. Optimal allocation without money: An engineering approach. *Management Sci.* **62**(4): 1078–1097.

Aziz, H., F. Brandt, M. Brill. 2013. The computational complexity of random serial dictatorship. *Econ. Lett.* **121**(3): 341–345.

Bapna, R., A. Barua, D. Mani, A. Mehra. 2010. Research commentary: Cooperation, coordination, and governance in multisourcing: An agenda for analytical and empirical research. *Inf. Syst. Res.* **21**(4): 785–795.

Bichler, M., J. K. Goeree. 2017. *Handbook of Spectrum Auction Design*. Cambridge University Press, Cambridge.

Birkhoff, G. 1946. Three observations on linear algebra. *Univ. Nac. Tucumán. Revista A* **5**: 147–151.

Bogomolnaia, A., H. Moulin. 2001. A new solution to the random assignment problem. *J. Econ. Theory* **100**(2): 295–328.

Bowman, E. H. 1963. Consistency and optimality in managerial decision making. *Management Sci.* **9**(2): 310–321.

Brandt, F., J. Hofbauer, M. Suderland. 2017. Majority graphs of assignment problems and properties of popular random assignments. In *Proceedings of the 16th Conference on*

Autonomous Agents and MultiAgent Systems, International Foundation for Autonomous Agents and Multiagent Systems, pp. 335–343.

Budish, E. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *J. Polit. Econ.* **119**(6): 1061–1103.

Budish, E., J. Kessler 2021. Can market participants report their preferences accurately (enough)? *Management Sci.* Forthcoming.

Budish, E., Y.-K. Che, F. Kojima, P. Milgrom. 2013. Designing random allocation mechanisms: Theory and applications. *Am. Econ. Rev.* **103**(2): 585–623.

Budish, E., G. P. Cachon, J. B. Kessler, A. Othman. 2017. Course match: A large-scale implementation of approximate competitive equilibrium from equal incomes for combinatorial allocation. *Oper. Res.* **65**(2): 314–336.

Burke, E. K., P. De Causmaecker, G. V. Berghe, H. Van Landeghem. 2004. The state of the art of nurse rostering. *J. Sched.* **7**(6): 441–499.

Che, Y.-K., F. Kojima. 2010. Asymptotic equivalence of probabilistic serial and random priority mechanisms. *Econometrica* **78**(5): 1625–1672.

Clarke, E. 1971. Multipart pricing of public goods. *Public Choice* **11**: 17–33.

Diamant, A., J. Milner, F. Quereshy. 2018. Dynamic patient scheduling for multiappointment health care programs. *Prod. Oper. Manag.* **27**(1): 58–79.

Diebold, F., M. Bichler. 2017. Matching with indifferences: A comparison of algorithms in the context of course allocation. *Eur. J. Oper. Res.* **260**(1): 268–282.

Diebold, F., H. Aziz, M. Bichler, F. Matthes, A. Schneider. 2014. Course allocation via stable matching. *Bus. Inf. Syst. Eng.* **6**(2): 97–110.

Ehlers, L., B. Klaus. 2003. Coalitional strategy-proof and resource-monotonic solutions for multiple assignment problems. *Soc. Choice Welfare* **21**(2): 265–280.

Fischer, G. W. 1972. Four methods for assessing multi-attribute utilities: An experimental validation. Technical report, Michigan University Ann Arbor Engineering Psychology Lab.

Gale, D., L. S. Shapley. 1962. College admissions and the stability of marriage. *Am. Math. Mon.* **69**(1): 9–15.

Gibbard, A. 1977. Manipulation of schemes that mix voting with chance. *Econometrica* **45**: 665–681.

Goetzendorff, A., M. Bichler, B. Day, P. Shabalin. 2015. Compact bid languages and core-pricing in large multi-item auctions. *Management Sci.* **64**(7): 1684–1703.

Groves, T. 1973. Incentives in teams. *Econometrica* **41**: 617–631.

Hanley, J. A., B. J. McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* **143**(1): 29–36.

Heydenreich, B., R. Müller, M. Uetz. 2007. Games and mechanism design in machine scheduling—an introduction. *Prod. Oper. Manag.* **16**(4): 437–454.

Jung, K. S., M. Pinedo, C. Sriskandarajah, V. Tiwari. 2019. Scheduling elective surgeries with emergency patients at shared operating rooms. *Prod. Oper. Manag.* **28**(6): 1407–1430.

Kamada, Y., F. Kojima. 2017. Recent developments in matching with constraints. *Am. Econ. Rev.* **107**(5): 200–204.

Karaenke, P., M. Bichler, S. Minner. 2019. Coordination is hard: Electronic auction mechanisms for increased efficiency in transportation logistics. *Management Sci.* **65**(12): 5884–5900.

Karaenke, P., M. Bichler, S. Merting, S. Minner. 2020. Non-monetary coordination mechanisms for time slot allocation in warehouse delivery. *Eur. J. Oper. Res.* **286**(3): 897–907.

Katta, A.-K., J. Sethuraman. 2006. A solution to the random assignment problem on the full preference domain. *J. Econ. Theory* **131**(1): 231–250.

Kavitha, T., J. Mestre, M. Nasre. 2011. Popular mixed matchings. *Theoret. Comput. Sci.* **412**(24): 2679–2690.

Kreipl, S., M. Pinedo. 2004. Planning and scheduling in supply chains: An overview of issues in practice. *Prod. Oper. Manag.* **13**(1): 77–92.

Krishna, A., M. U. Ünver. 2008. Research note: Improving the efficiency of course bidding at business schools: Field and laboratory studies. *Mark. Sci.* **27**(2): 262–282.

Kumar, S., V. Mookerjee, A. Shubham. 2018. Research in operations management and information systems interface. *Prod. Oper. Manag.* **27**(11): 1893–1905.

Lee, S. J., G. R. Heim, C. Sriskandarajah, Y. Zhu. 2018. Outpatient appointment block scheduling under patient heterogeneity and patient no-shows. *Prod. Oper. Manag.* **27**(1): 28–48.

Lehmann, D., R. Müller, T. Sandholm. 2006. The winner determination problem. P. Cramton, Y. Shoham, R. Steinberg, eds. *Combinatorial Auctions*. MIT Press, London, 297–318.

Lemay, B., A. Cohn, M. Epelman, S. Gorga. 2017. New methods for resolving conflicting requests with examples from medical residency scheduling. *Prod. Oper. Manag.* **26**(9): 1778–1793.

Li, S. 2017. Obviously strategy-proof mechanisms. *Am. Econ. Rev.* **107**(11): 3257–87.

Li, M. 2020. Ties matter: Improving efficiency in course allocation by allowing ties. *J. Econ. Behav. Organ.*, **178**: 354–384.

Manlove, D. F. 2013. *Algorithmics of Matching Under Preferences*. Series on Theoretical Computer Science. World Scientific Publishing Company Incorporated, Singaporee.

Mas-Colell, A., M. D. Whinston, J. R. Green, et al. 1995. *Microeconomic Theory*, Vol. **1**. Oxford University Press, New York.

May, J. H., W. E. Spangler, D. P. Strum, L. G. Vargas. 2011. The surgical scheduling problem: Current research and future opportunities. *Prod. Oper. Manag.* **20**(3): 392–405.

Müller, A., D. Stoyan 2002. *Comparison Methods for Stochastic Models and Risks*, Vol. **389**. Wiley, New York.

Nguyen, T., A. Peivandi, R. Vohra. 2016. Assignment problems with complementarities. *J. Econ. Theory* **165**: 209–241.

Othman, A., C. Papadimitriou, A. Rubinstein. 2016. The complexity of fairness through equilibrium. *ACM Trans. Econ. Comput.* **4**(4): 20.

Ou, J., V. N. Hsu, C.-L. Li. 2010. Scheduling truck arrivals at an air cargo terminal. *Prod. Oper. Manag.* **19**(1): 83–97.

Pápai, S. 2001. Strategyproof and nonbossy multiple assignments. *J. Public Econ. Theory* **3**(3): 257–271.

Pycia, M., P. Troyan. 2018. Obvious dominance and random priority. *SSRN Electron. J.* (2853563). https://doi.org/10.1145/3328526.3329613

Sampson, S. E. 2004. Practical implications of preference-based conference scheduling. *Prod. Oper. Manag.* **13**(3): 205–215.

Sönmez, T., M. U. Ünver. 2010. Course bidding at business schools. *Int. Econ. Rev.* **51**(1): 99–123.

Stewart, T. J. 1995. Simplified approaches for multicriteria decision making under uncertainty. *J. Multi-Criteria Decis. Anal.* **4** (4): 246–258.

Vickrey, W. 1961. Counterspeculation, auctions, and competitive sealed tenders. *J. Finance* **16**(1): 8–37.

Von Neumann, J. 1953. A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contr. Theor. Games* **2**: 5–12.

Yu, S., V. G. Kulkarni, V. Deshpande. 2020. Appointment scheduling for a health care facility with series patients. *Prod. Oper. Manag.* **29**(2): 388–409.

Zhou, L. 1990. On a conjecture by Gale about one-sided matching problems. *J. Econ. Theory* **52**(1): 123–135.

Zhou, S., Y. Ding, W. T. Huh, G. Wan. 2021. Constant job-allowance policies for appointment scheduling: Performance bounds and numerical analysis. *Prod. Oper. Manag.*. https://doi.org/10.1111/poms.13362

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article.

Appendix A. Figures.
Appendix B. The Lottery of the Summer Term Instance.
Appendix C. Rank Profiles with Truncation.
Appendix D. Instructions for Lab Experiments.