The Quest Draft: an Automated Course Allocation Algorithm

Richard Hoshino and Caleb Raible-Clark

Quest University Canada, Squamish, British Columbia, Canada

Abstract

Course allocation is one of the most complex issues facing any university, due to the sensitive nature of deciding which subset of students should be granted seats in highly-popular (market-scarce) courses. In recent years, researchers have proposed numerous solutions, using techniques in integer programming, combinatorial auction design, and matching theory. In this paper, we present a four-part AI-based course allocation algorithm that was conceived by an undergraduate student, and recently implemented at a small Canadian liberal arts university. This new allocation process, which builds upon the Harvard Business School Draft, has received overwhelming support from students and faculty for its transparency, impartiality, and effectiveness.

Introduction

In the *multi-unit assignment problem*, a set of indivisible objects is to be allocated amongst a set of individuals, where individuals have preferences over bundles of objects, and monetary transfers are forbidden. The goal is to design a mechanism where individuals are incentivized to report their preferences truthfully, with the final allocation of objects to individuals maximizing overall welfare. Real-life applications include the assignment of shifts to workers, players to sports teams, shared lab resources to scientists, and takeoff-and-landing slots to airlines (Budish and Cantillon 2012).

Course allocation is the most-studied application of the multi-unit assignment problem, and is a well-known open problem in market design theory (Budish 2011). At most post-secondary institutions, it is impossible for all students to receive their most preferred set of courses, due to limits on class size and other scheduling constraints. University administrators are thus tasked with the challenging problem of matching students to courses in a fair and equitable way.

Many business schools use auction-based mechanisms, even though these mechanisms lead to welfare loss and are easily manipulable (Sonmez and Unver 2010). Some large universities allow students to register for courses sooner if they have a higher grade point average, which ensures that those with the highest GPAs get into all the best courses. This unbalanced mechanism also reduces overall welfare.

Over the years, economists have played a key role in designing welfare-maximizing strategy-proof solutions to single-unit assignment problems; among the numerous success stories are matching medical school graduates to hospitals, and matching high school students to local public schools (Roth 2002). However, the multi-unit assignment problem is much more challenging, and recent breakthroughs have relied on AI-based techniques. One recent example is the methodology to calculate nearly-optimal market-clearing prices for each course, using a mixed-integer program (Budish, Othman, and Sandholm 2010).

The authors of this paper, a faculty member and an undergraduate student at a small Canadian liberal arts university, were invited to propose a new course allocation mechanism, to replace the previous process where students were allocated their entire semester of courses on a first-come-first-served basis. Many alternatives were considered, including the auction-based MIP described above; ultimately, the authors recommended a process that was less of a "black box" – a mechanism that was effective and equitable, transparent and easy-to-understand. Our proposed mechanism, the *Quest Draft*, was implemented in August 2013.

In this paper, we present our four-part algorithm that builds upon the draft used at the Harvard Business School: (i) a cyclically-shifted "snake" order that optimally spreads out draft picks among the students; (ii) a backtracking system where each student's final schedule is guaranteed to meet pre-specified caveats (e.g. student x gets course c_i if and only if she also gets c_j); (iii) a pairwise-swapping process that iteratively switches the timeslots of two courses to minimize the number of instances when students are "shafted" by having no available course to choose from; (iv) and a perfect-information computer proxy that is programmed to strategically draft courses on behalf of each student, based on the course preferences of all n students.

This paper proceeds as follows: first, we describe the unique features of our university that made the previous process of course allocation so problematic. We then detail each of the four parts of our algorithm, explain how we implemented the first three components for the 2013-2014 academic year, and share the results. We then explore the potential complications of implementing the final part of our algorithm, discuss its strengths and limitations, and conclude the paper with avenues for further research.

Context

Quest University Canada, a small liberal arts college in Squamish, British Columbia, opened its doors in September 2007. The founders of Quest designed a curriculum where students spend their first two years engaged in a rigorous Foundation program spanning the arts and sciences, after which the students have the opportunity to drive their own education by posing a unique "Question" that directs the final two years of their studies.

Each student's Question can be thought of as their major. As each Question is expected to be interdisciplinary, students are required to take upper-year courses across multiple divisions (Life Sciences, Humanities, etc.) Thus, each of the 550 students has a unique self-designed set of courses.

At many large undergraduate institutions, there are few capacities or restrictions on course enrollment; students sign up for their preferred courses, and a classroom is selected based on the number of students that register for a particular course. This flexibility does not exist at Quest, where all classes are capped at 20 students.

To further complicate matters, every class at this university operates under the *block format*, where students take just one course at a time for an entire month, with three hours of class time each day. Unlike other universities where there are dozens of timeslots for various courses from early in the morning to late in the evening, at Quest there are only four feasible time slots – a fixed time slot for each month of the semester. Thus, in this context, the words "block" and "month" are synonymous with "time slot".

Because there are only four time slots each semester, students often experience the frustration of seeing two Question-related courses offered in the same block, forcing a student to pick one or the other. While this is not a problem at larger institutions, where multiple sections of a course are offered each semester and a student can select any convenient time slot, this is impossible at Quest, where a course such as Calculus II is offered only once every academic year.

No master course schedule will satisfy every Quest student, due to these self-designed majors, the 20-person course cap, the block format, and the scarcity of course offerings.

Inevitably, certain courses are more popular than others, and are sought after by several dozen students. To address this issue and be as "fair" as possible, Quest had, since its inception, registered students for courses on a first-come-first-served basis. Each semester's registration period would open at 4:30PM on a particular day; students would register online for their four blocks by selecting their four desired courses that semester and registering with a single click.

In the language of market design theory, this process was known as a *Random Serial Dictatorship* (Abdulkadiroglu and Sonmez 1998), where the students who clicked their mouse buttons first got into *all* four of their desired courses, while other students couldn't enroll in any of their highly-ranked courses because in each block, the 20^{th} student clicked the mouse button a few seconds before them. Often, the most popular courses were completely full by 4:31PM. This course registration mechanism, the "Fastest-Finger" Serial Dictatorship, rewarded the students who knew the precise location on campus with the fastest Wi-Fi connection.

In August 2013, the senior administration approved the Quest Draft, an AI-inspired approach to course allocation, to replace the Fastest-Finger Serial Dictatorship that had caused anxiety among the student body, and was widely viewed as unfair. We now describe our four-part algorithm, used to register students for the Spring 2014 semester.

Part 1: Cyclically-Shifted Snake-Order Draft

Instead of a process by which students choose their entire bundle of courses with a single click, Quest students now register each semester through an automated four-round draft, where students submit a pre-determined "wishlist" of 20 desired courses over all four months, in decreasing order of preference. In each round, every student is given one selection, and is assigned their top-choice course from all the courses for which there is still an open seat.

Once student x has successfully enrolled in course c (e.g. *Evolution* in March), we update that student's wishlist. First, we remove every other course ranked that month, since student x can only take one course in each month. Second, if student x has ranked the same course offering in a different block (e.g. *Evolution* in January), then we delete that choice from their wishlist. As soon as course c has reached 20 students, we eliminate c from every student's wishlist.

The automated draft program was written in both Python and Maplesoft; the actual Maplesoft code is provided below:

```
for n to nops(draftorder) do
 x := draftorder[n]:
 c := wishlist[x][1]:
 enrollment[c] := enrollment[c]+1:
 classlist[c] := classlist[c],x:
 schedule[x] := schedule[x],c:
 wishlist[x] := update(wishlist[x],c,x):
 if enrollment[c] = courselimit[c] then
  for l to rows do z := inputmatrix[l,1]:
   wishlist[z] := remove(wishlist[z],c):
  end do:
 end if:
end do:
```

In the past, registration took place over a period of four days, starting with the 4^{th} years on a Monday and ending with the 1^{st} years on that Thursday. (And within each year, the course allocation mechanism was the Fastest-Finger Serial Dictatorship, starting at 4:30PM.) Through this automated draft program, registration no longer takes four days: with 550 students each taking four courses, the Quest Draft has 2200 picks, with a total runtime of just 1.6 seconds.

A natural question is how we should decide the draft order. The National Hockey League (NHL) uses a ladder draft, where the team that gets the first pick in round i gets the first pick in round $i+1$. On the other hand, Harvard uses a snake-order draft, where the student who gets the last pick in Round i gets the first pick in Round $i+1$. We implemented a *cyclically-shifted* snake-order draft: for example, with $n = 8$ students, the draft order is as follows.

Round 1: A, B, C, D, E, F, G, H
Round 2: H, G, F, E, D, C, B, A
Round 3: E, F, G, H, A, B, C, D
Round 4: D, C, B, A, H, G, F, E

We first randomly permute the n students to decide the draft order for Round 1. Once Round 1 is decided, all subsequent rounds follow a fixed pattern, which we now describe.

Let $p(i) := (p_1(i), p_2(i), p_3(i), p_4(i))$, where $p_j(i)$ denotes the pick number of student i in round j. In our example with $n = 8$, the first student would have the four-tuple $p(1) = (1, 8, 5, 4)$. In general, if there are $n = 4k$ students, then $p(i) = (i, 4k{+}1{-}i, i{+}2k, 2k{+}1{-}i)$ if $1 \leq i \leq 2k$ and $p(i) = (i, 4k+1-i, i-2k, 6k+1-i)$ if $2k+1 \leq i \leq 4k$.

If $n \not\equiv 0 \pmod 4$, then we just add dummy students to make n a multiple of 4. It is easy to show that this cyclically-shifted snake-order is the only draft ordering that satisfies each of the following four "equitability" properties:

(a) Each student's picks are spread out as evenly as possible, with exactly one pick in each of the four quartiles.

(b) The sums $p_1(i) + p_2(i)$ and $p_3(i) + p_4(i)$ are constant for each $1 \leq i \leq n$.

(c) $p_1(i) < p_2(i)$ if and only if $p_3(i) > p_4(i)$.

(d) The value $\min_i\{p_1(i)^2 + p_2(i)^2 + p_3(i)^2 + p_4(i)^2\}$ is as small as possible.

Consider a scenario where 100 fourth-year students register for a semester, where each course is capped at twenty students and exactly five courses are offered in each block. In each of the four months, the 100 students rank the five offered courses in order of their personal preference, and (naturally) hope to get into the course that is their first choice.

We ran hundreds of simulations where each student's wishlist of 20 courses was randomized, and compared the results of the Random Serial Dictatorship (RSD) with the Cyclically-Shifted Snake-Order Draft (CSSOD). Each student's score is the sum total of preference rankings over all four blocks. The best possible score is $1 + 1 + 1 + 1 = 4$ points, which occurs whenever a student gets into all of their first-choice courses; the worst possible score is $5 + 5 + 5 + 5 = 20$ points. One comparison is given in Figure 1 below.

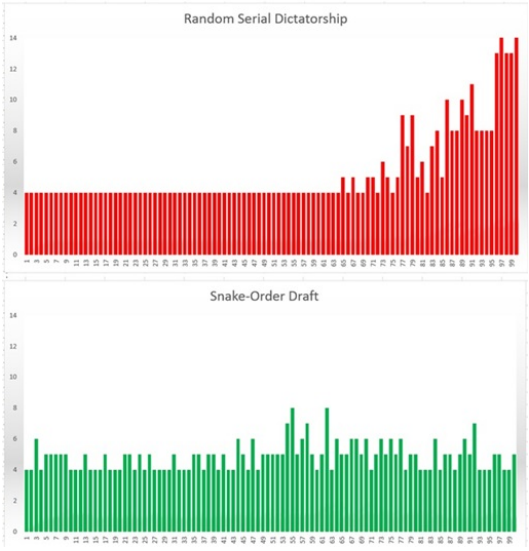

Figure 1: Comparison of the RSD and the CSSOD

In Figure 1, the average score for the RSD is 5.27 points, with only 81% of students getting into at least three of their first-choice courses. However, in the CSSOD, the average score is 4.84 points, with 98% of students getting into at least three of their first-choice courses. Analyzing all of our simulation results, we discover that the average score of the CSSOD is about 5% less than the average score of the RSD.

It is clear that the RSD is a sub-optimal system; equitability is lacking when fourth-year student x gets to choose all of his courses before fourth-year student y. Total welfare loss also follows intuitively, since student x's last-choice course might be student y's first-choice course. This intuition, formally known as a *negative callousness externality*, has been confirmed in an analysis of the Harvard Business School draft, where "the lucky gain less than the unlucky lose." (Budish and Cantillon 2012).

As a result, the university has replaced the four-day Serial Dictatorship with the automated Cyclically-Shifted Snake-Order Draft, which we run four straight times, once for each of the four years. (Thus, all fourth-year students pick all four of their courses before any third-year student makes a single pick.) The senior administration requested this policy because upper-year students at Quest have fewer options of courses to take in any given block, due to their self-built major's requirements, and have a more urgent need to complete certain courses that are required for graduation.

Part 2: Caveats and Backtracking

In a sports league draft, there is a substantial time window between picks, which enables each team to adapt their strategy based on which players have been selected thus far. A football team may have every intention of drafting a strong quarterback in Round 1 and a decent wide receiver in Round 2, but may choose to flip that order if an elite top-rated wide receiver is unexpectedly up-for-grabs in Round 1.

Unlike the National Football League draft which takes place over a three-day period, our 2200-pick course allocation draft runs in less than two seconds. To ensure that students don't end up with an undesirable bundle of courses, we allow students to create special "caveats", i.e., personal constraints that apply only to them.

These caveats take on two forms: an *at-most* condition that guarantees that the student doesn't get any more than r courses out of a pre-specified set $\{c_1, c_2, \ldots, c_t\}$, as well as an *if-and-only-if* condition that guarantees that the student gets course c_i iff she also gets course c_j. The first caveat is known as a *preference constraint*, and can be modeled using a bihierarchical structure (Budish et al. 2013).

These caveats are crucial in our course allocation mechanism; without it, students could end up with a undesirable combination of courses, or worse, end up with a situation that actually occurred at the University of Michigan, where students were incorrectly registered for multiple sections of the same course (Krishna and Unver 2008).

Thanks to these caveats, students are empowered with more options and flexibility. For example, a first-year student who wishes to take exactly one upper-level Humanities course could rank all r such courses among her first r selections while invoking the at-most caveat.

The at-most caveat is easy to program: for example, suppose student x has requested no more than two courses from the set $\{c_1, c_2, c_3, c_4, c_5\}$. Suppose x drafts c_1 in Round 1 and c_2 in Round 2. As soon as c_2 has been picked, courses c_3, c_4, and c_5 are automatically eliminated from x's wishlist.

The inaugural Quest Draft occurred in November 2013, and was used to register students for the Spring 2014 semester, covering January to April. Of the 550 students who took part in the draft, about half requested an at-most caveat; however, the if-and-only-if caveat was only invoked by a handful of students, *all* of whom used it to request two consecutive blocks of a foreign language – e.g. "Spanish 1 in March if and only if Spanish 2 in April". (Every student must complete two foreign language courses, and the most popular option is for students to do this in consecutive months during some semester of their time at Quest.)

To handle the if-and-only-if caveat, we employ a special *backtracking* mechanism that works as follows: suppose student x has c_1, c_2, c_3 as the first three courses on her wishlist, and also specifies the caveat "c_1 iff c_2". Suppose x picks c_1 in Round 1, and before she makes her Round 2 pick, the final seat in c_2 is taken. Then we *backtrack* to x's original first-round selection, reset all the selections that have been made after this pick, eliminate courses c_1 and c_2 from x's wishlist, and have her draft c_3, the first course on her updated wishlist. We then re-start the rest of Round 1, starting with the student picking right after x.

This backtracking step would of course be unnecessary if we could figure out in advance that x would not be able to register for c_1 and c_2; we would only have to run the entire draft once, as long as we first remove these two courses from x's original wishlist. However, we found complex instances involving multiple individuals where it wasn't obvious which if-and-only-if caveats would be triggered. Therefore, we opted for the more conservative method of backtracking. While this approach takes longer to run, the algorithm is still polynomial-time and is guaranteed to terminate.

Theorem 1 *The Quest Draft is a deterministic polynomial-time course allocation algorithm.*

Proof Let there be n students, each ranking $m = O(n)$ courses on their wishlist. There are $4n$ picks in the draft, since each student is enrolled in four courses per semester. Even if we assume that each course has an enrollment cap of one student, every draft pick requires at most $O(nm) = O(n^2)$ operations to update each student's wishlist and remove the drafted course that is no longer available, as well as check the at-most caveats to verify if any of the m courses need to be removed from that student's wishlist. Thus, at worst, there are $4n \times O(n^2) = O(n^3)$ picks required to run the entire draft, assuming no backtracking.

Since student x ranks m courses on her wishlist, the draft will backtrack on x's behalf at most $\frac{m}{2}$ times, since two courses are removed from x's wishlist for each backtrack. (Clearly, an infinite loop cannot occur, as the cardinality of x's wishlist drops after each backtrack.) Thus, the number of total backtracks is at most $\frac{mn}{2} = O(n^2)$. Therefore, the total running time of our course allocation algorithm is, at worst, $O(n^3) \times O(n^2) = O(n^5)$. ∎

Naturally, the running time is far better than $O(n^5)$ in practice. We noted that with $n = 550$ students and $4n = 2200$ picks, the entire course allocation draft ran in just 1.6 seconds, including caveats. And there were only three backtracks, far less than the theoretical maximum of $O(n^2)$.

As mentioned before, the if-and-only-if caveat was only specified by students wishing to take two consecutive blocks of a foreign language. In the Spring 2014 semester, six language blocks were offered: French 1, French 2, Chinese 1, Chinese 2, Spanish 1, and Spanish 2.

Three first-year students were backtracked after they requested "French 1 iff French 2". All three of these students listed these two courses with their first two picks, and so the number of reset picks was minimal. Neither Chinese course filled up to 20 students, so backtracking wasn't necessary. Finally, Spanish 2 had exactly 20 students after all the second-year students had finished drafting their four courses. Thus, no backtracking was necessary in this scenario: any first-year student requesting "Spanish 1 iff Spanish 2" had both courses removed from their wishlist.

Part 3: Pairwise-Swapping Algorithm

When Quest allocated courses via the Fastest-Finger Serial Dictatorship, students were averse to registering for less-popular courses in case they were later cancelled due to low enrollment. A few days after everyone had registered, the senior administration would decide if any courses would be cancelled, with the typical threshold being fewer than 5 students. The unlucky students who had registered for cancelled courses would then have to sign up, on a first-come-first-served basis, for whatever courses still had an open seat.

This policy was especially problematic for upper-year students, who would be informed of a cancelled course a week after they had signed up it, and then discover that the remaining seats in their second- and third-choice courses that month were already taken by first-year students. There was no mechanism to "bump out" first-years in favour of these unlucky upper-years, who should have been given priority.

As a result, many students interested in taking a less popular course c_i would instead register for the more popular option c_j in the same month to grab one of the coveted twenty seats; once the university announced that c_i would indeed proceed as planned, these students would drop c_j and take one of the open seats in c_i. In implementing our course allocation mechanism, steps were taken to prevent this type of strategic (mis)-behaviour, which had the added drawback of cancelling classes that should have run had students been honest in reporting their preferences.

The first fix was to decide which classes would be cancelled *prior* to finalizing the results. This simple change incentivized students to rank the courses they actually wanted, so that these desired courses would run as intended.

When we ran the draft, we discovered that only one course was allocated to fewer than five students. After this course c was cancelled by the Academic Dean, we simply re-ran the draft with course c off everybody's wishlist, as if c never existed in the first place. Thus, the four students who were matched to this course were not penalized for honestly ranking c in their 20-course wishlist.

Because each iteration of the Quest Draft runs in 1.6 seconds, we had much time to experiment before the final allocation of students to courses was officially published. Much to our surprise, a large percentage of fourth-years registered for required "Foundation" courses in February and March, which they had apparently not yet taken. As the university had underestimated the number of Foundation courses that students would take in February and March, many first-years and second-years were *shafted*, with no available course to take in either of these two months.

Many shafts were unavoidable, as students poorly selected the 20 courses on their wishlist. For example, a second-year student ranked three highly-popular January courses, with no other options that month. For each of these three courses, the twenty seats were drafted by third-years and fourth-years, leaving her with no course to choose from in January. But in other scenarios, students were shafted through no fault of their own; there were simply not enough seats in Foundation-level courses in February and March.

To address this issue, we added a "pairwise-swapping" algorithm to our computer program, a simplified variation of a discrete optimization technique known as *simulated annealing* that has solved complex problems in sports tournament scheduling (Kendall et al. 2010). This heuristic determines the effect of iteratively switching the months that two courses are offered, with the goal of minimizing the total number of shafted students:

(a) Let TA be the set of teaching assignments for the k professors: each professor p has the teaching schedule $\{c_1, c_2, c_3, c_4\}$ for the four months, where at least one c_i is empty (representing a non-teaching block).

(b) For each of the $\binom{4}{2}k = 6k$ possible triplets (p, c_i, c_j), we check whether the scenario of professor p swapping the months of these two courses would lead to fewer students being shafted.

(c) For the triplet (p, c_i, c_j) that best reduces the number of total shafts, we swap c_i and c_j in professor p's schedule, reflect this change in TA, and go back to step (a).

(d) End when none of the $6k$ triplets (p, c_i, c_j) yields an improvement, and output the teaching assignment TA.

This simple pairwise-swapping heuristic, recently developed by one of the authors to develop best-known bounds for an open problem in scheduling theory (Goerigk et al. 2014), can be applied in this context to find a local optimum in each iteration. We found that this approach tended to be particularly effective when some c_i was empty (i.e., professor p had a non-teaching block in month i). As all these courses are unique, each with its own professor, we made the assumption that a student wanting to take course c in one month would be equally content to take c in another month.

We discovered a simple scenario that would significantly reduce the total number of shafts, with one professor moving her course from January to February, and another moving his course from April to March. Both professors agreed to shift their non-teaching block by one month, and as a result, eighteen first-year students were no longer shafted, having been allocated a Foundation course in February and/or March.

Of course, there were other teaching assignments that led to fewer shafts, but they were not possible to implement, mostly due to professors having pre-arranged commitments during their non-teaching block.

This "pairwise-swapping" algorithm led to a major improvement at minimal cost; however, this simple solution would have been impossible to find had the university not replaced the Fastest-Finger Serial Dictatorship. Only 38 of the 550 students were shafted, all in exactly one month, with nearly every shaft being unavoidable due to how these students filled out their 20-course wishlist. (Once the draft results were published, the Registrar asked these 38 first-year and second-year students to sign up for any available course that month, on a first-come-first-served basis.)

Part 4: Strategic Computer Proxy

In order to register students for the Spring 2014 semester, we ran the Quest Draft in November 2013, with each of the 550 students providing a ranked wishlist of 20 courses, among the 119 courses that were offered between January and April. In implementing this course allocation mechanism, we paid much attention to its features, from the cyclically-shifted snake ordering of draft picks, the flexibility of caveats, and the pairwise-swapping that enabled more students to be allocated to courses. However, there was one design weakness that we hoped would not be discovered and exploited by students. Alas, the flaw was discovered, and several students took full advantage.

In *any* draft-based multi-unit allocation mechanism, the strategic manipulation of preferences can lead to sub-optimal outcomes. To illustrate this principle, suppose that x wishes to take course c_i in month i, for $1 \leq i \leq 4$, and suppose that x's true preference ranking is $\{c_4, c_3, c_2, c_1\}$. From talking with fellow students, x concludes that c_4 in April will not be selected by 20 students, and so she under-reports her preference for c_4, believing she can draft this course in Round 4. Conversely, x learns that c_1 in January is the most popular course at the university, and so she over-reports her preference for c_1, hoping to draft this course in Round 1. As a result, x's reported ranking is $\{c_1, c_3, c_2, c_4\}$.

From post-draft interviews with Quest students, we learned that some students under-reported desired courses and turned out not to get them, and over-reported less-desired courses which harmed others who strongly desired them. These sub-optimal outcomes, known as *ex-post Pareto inefficiencies*, occur when students are not truthful in ranking their preferences. At Harvard Business School, which has run their draft since the mid-1990s, researchers have demonstrated that the draft is manipulated in practice and that these manipulations cause significant welfare loss. A detailed analysis reveals that nearly half of Harvard students are unambiguously harmed by strategic play, whereas only 10 percent benefit (Budish and Cantillon 2012).

Ideally, we desire a mechanism that is *strategy-proof*, where every student's optimal strategy is to report his or her preferences truthfully. While our mechanism is obviously strategy-proof if students are only selecting a single course in a fixed month (the single-unit assignment problem), it is

not if students are selecting four courses across four different months (the multi-unit assignment problem).

A well-known result in market design theory states that dictatorships are the *only* multi-unit assignment mechanisms that are strategy-proof and ex-post Pareto efficient (Pápai 2001). But as we saw earlier in the paper, mechanisms such as the Random Serial Dictatorship (RSD) lead to a clear loss of welfare, as well as the ex-ante perception of unfairness.

To address these weaknesses and mitigate errors in strategic play, we proposed a *proxy draft*, where a single agent (known as a "perfect-information proxy") possesses the course rankings of all n students, and drafts strategically on behalf of each student based on how many seats are left in each course desired by that student. If the proxy ever makes a mistake in drafting a less-desired more-popular course c_i before a more-desired less-popular course c_j, we *backtrack*.

Our proxy algorithm is simple to describe: for each of the $4n$ picks in the draft, consider that student's top-choice pick in every month that she has not yet been allocated a course. Then the proxy will draft, on behalf of that student, the available course with the fewest number of open seats. (If there is a tie between two or more courses, the strategic proxy will draft the course appearing highest on that student's wishlist.)

Let us illustrate the proxy algorithm with our earlier example, where student x has $\{c_4, c_3, c_2, c_1\}$ as the first four courses on her 20-course wishlist, with c_i offered in month i. Suppose that when x makes her first-round selection, all four of these courses are still available, and c_1 is, as expected, the course with the fewest number of available seats. Then the proxy will draft c_1 on behalf of x in Round 1.

The proxy's goal is to draft all four of these courses on x's behalf. However, suppose that c_2 unexpectedly reaches full capacity before x makes her second-round pick. Since x prefers c_2 to c_1, the algorithm backtracks to x's original Round 1 selection, ignores x's top-choice course in January, and selects the most popular course among $\{c_2, c_3, c_4\}$. Without loss of generality, suppose this course is c_2.

Suppose x drafts c_2 in Round 1, c_3 in Round 2, and before her third-round pick, c_4 reaches full capacity. We then backtrack to x's first-round selection, have her draft c_3 instead, and re-continue the draft from there. (She will then select c_4 with her second-round pick.) At the conclusion of the draft, suppose x has selected $\{c_3, c_4, d_2, d_1\}$ with her four picks, with courses d_2 and d_1 appearing 8^{th} and 19^{th} on her original 20-course wishlist.

For each student, define $f(k) = 2^{-k}$ to be the *utility* of receiving the k^{th} course on her wishlist. Then in the above example with x, her total utility is $\frac{1}{2^2} + \frac{1}{2^1} + \frac{1}{2^8} + \frac{1}{2^{19}}$. Because of the backtracking feature, the perfect-information proxy maximizes the total utility of each student in the draft, given the submitted course rankings of all n students.

Our proposed utility-maximizing backtracking proxy is similar but not equivalent to the Proxy Bidding Mechanism (PBM), a recently-developed approach that matches students to courses using a simultaneous ascending auction, where proxies bid for courses using an artificial currency (Kominers, Ruberry, and Ullman 2010).

In the PBM, each of the n students is given four bills representing different bid values, with all bid values distinct and being an integer between 1 and $4n$ inclusive. Like in our cyclically-ordered snake-order draft, each student is given four bills so that the sum total of "dollars" is constant, equaling $8n + 2$ for each student.

The student then gives these four bills, along with a strictly-ordered list of desired courses, to a proxy. The algorithm then opens up the market for bidding. Each proxy bids the lowest possible value for their most-desired course. Bids can be increased, but not decreased. Bidding continues until no proxies wish to make further bids.

The PBM is effective because it is resilient to the strategic play seen in the Harvard Business School draft, where students rank more-desirable classes higher than their preference for them, in order to claim spots earlier in the draft. Consequently, net utility has been shown to be greater in the PBM than in the Harvard draft.

In the PBM, lowest-value bids are made first (i.e., the person with the 1-dollar bill makes the opening bid), whereas in the Quest Draft, the first pick is given to the individual holding the $4n$-dollar bill.

For this reason, it is likely that our backtracking proxy is preferable to the PBM in instances where demand is highly concentrated in a small number of courses, which is precisely the situation at our university. The PBM would involve a series of increasing bids cycling through each proxy, whereas our approach would involve several early uncontestable bids, and reduce overall run time.

The PBM includes "Extended Preference" features: *IF*, which assigns a given less-desirable course only if a given more-desirable course has been assigned; and *NOTIF*, which assigns a given less-desirable course only if a given more-desirable course has not been assigned. A student wishing to take only one of $\{c_1, c_2, c_3\}$, in this order, would indicate this by: [c_2 *NOTIF* c_1, c_3 *NOTIF* c_1, and c_3 *NOTIF* c_2].

On the other hand, our four-part algorithm includes a simple *at-most* caveat, which collapses three statements down to one. Furthermore, a student requesting a complicated constraint such as "give me no more than three of the following seven courses" can do so easily with a single *at-most* caveat, rather than a complex chain of conditional *IF* and *NOTIF* statements.

Unlike the PBM, we included the *if-and-only-if* caveat because it is more comprehensive, preserving a student's ability to create two-way conditional registration, especially in situations where one wishes to take back-to-back language blocks, and wants either both courses or neither. Clearly, both of our caveats are compatible with the strategic proxy: if any caveat has been violated, we simply backtrack.

Decision and Deployment

We recommended this deterministic polynomial-time proxy algorithm for two reasons: first, by having the proxy play on each student's behalf, we would eliminate errors from incorrect strategic under-reporting; second, we would level the playing field, allowing the proxy to utilize the student's *realized position* in the drafting order, thus increasing ex-ante welfare relative to what actually happens in a pick-by-pick draft (Budish and Cantillon 2012).

When we proposed the strategic proxy to focus groups of students and faculty, it was rejected: in addition to its perception of being too complicated, the proxy draft would unfairly guarantee the advantage of students "majoring" in a low-demand subject (e.g. physics) who wished to take electives in a high-demand subject (e.g. psychology).

Furthermore, this algorithm would maximize each student's total utility *assuming* a geometric utility function such as $f(k) = 2^{-k}$. However, this is not the correct utility function as most students would prefer the four courses ranked $\{2, 3, 4, 5\}$ on their wishlist over the four courses ranked $\{1, 18, 19, 20\}$.

For these reasons, the perfect-information strategic proxy was not included in the Quest Draft. Nevertheless, we successfully implemented the first three parts of our algorithm, and replaced the Fastest-Finger Serial Dictatorship.

After the new course allocation policy was approved by the senior administration, the authors spent the first two weeks of the Fall 2013 semester running information sessions, to explain the new process to all students and faculty. In administering the Quest Draft, we collected student preferences with an online survey, which exported an Excel spreadsheet readable by our Maplesoft program.

Once the Academic Dean approved the final teaching assignment of professors to courses (the output of our pairwise-swapping algorithm), we officially ran the 2200-pick draft to match students to these courses, with the output being a simple-to-read Excel file. The Quest Draft ran in just 1.6 seconds, on a stand-alone laptop with 2.75 GB main memory and a single 2.10 GHz processor.

In analyzing the results, we gave each student a score, the sum total of preference rankings over all four blocks. Every fourth-year student received the best possible score of $1 + 1 + 1 + 1 = 4$ points, as 100% of fourth-year students got into their first-choice course in each of the four months. The average scores were 4.48 for the third-years, 5.98 for the second-years, and 7.52 for the first-years. Of the third-year students, 89% got into their first-choice course in at least three of the four months.

As hoped for, we were able to distribute highly-popular courses among students in a more equitable way, to avoid the situation from previous years when there were extreme winners and losers.

Maintenance

While the Quest Draft ran flawlessly, the actual process of course registration was far more complicated. Given that there is no batch input method for course registration in the university's record-management system, the Registrar's Office had to manually input student-to-course assignments, all 2200 of them, one at a time using a drop-down menu.

This tedious input time (totaling 13 hours) was a poor use of human resources, and caused students to wait several days to discover which courses they had been allocated.

Ideally, there would be a simple way to upload our Excel file with the draft results directly into the university's record-management system. We are investigating ways to make this process much more efficient in the future.

The Maplesoft program is maintained by one of the authors of this paper (the faculty member), though it is officially housed in the Registrar's Office. A minor update is required in every registration cycle, to reflect the slate of courses that are offered each semester. However, the actual code used to run the first three parts of our algorithm will not require changing, and will remain robust as long as the university keeps operating under the *block format*, where students take just one course at a time for an entire month.

Conclusion

As shown in previous research, any proxy algorithm will over-report popular courses, and will not eliminate all costs of strategic behaviour. At Harvard, optimal welfare (predictably) occurs when all students honestly rank their course preferences. This is followed by the strategic proxy, then the pick-by-pick draft, with the random serial dictatorship being the worst mechanism (Budish and Cantillon 2012).

Our proposed perfect-information strategic proxy ensured an output that would maximize each student's total utility assuming a geometric function such as $f(k) = 2^{-k}$. While this is not the correct utility function for some students, this is indeed correct for those students who would prefer their first-choice course in a semester, compared to receiving the four courses ranked $\{2, 3, 4, 5\}$ on their wishlist.

Moving forward, one option is to have students indicate which utility function they prefer, which would of course be expressed in common language rather than as a mathematical formula. Then the backtracking feature of our strategic proxy algorithm would need to be rewritten to take different utility functions into account.

Alternatively, we could develop a different type of proxy algorithm. One promising approach is to model the draft as a finite repeated game with perfect information, with each individual proxy selecting a strategy corresponding to the game's Nash equilibrium (Kalinowski et al. 2013). This is a technique that we will carefully study in the months ahead.

Being at a small school with little to no bureaucracy, we are fortunate to have the freedom to experiment with novel approaches, with the full support of our administration. Our hope is that every iteration of the Quest Draft will lead to a more effective and efficient course allocation process for both the students and the Registrar's Office.

We see our university as a natural laboratory for testing cutting-edge ideas, and we welcome collaborative partnerships from those in the AI community, especially from researchers studying the multi-unit assignment problem.

We look forward to the day that AI research can lead to a welfare-maximizing strategic proxy that will optimize the process of allocating courses to students: not just at Quest University Canada, but at other universities throughout North America, and throughout the world.

Acknowledgements

We are grateful for the support of our community at Quest, most notably: David Helfand (President), Ryan Derby-Talbot (Chief Academic Officer), Tim Schoahs (Registrar), and Rich Wildman (Chair of the Summer Fellows Program).

References

Abdulkadiroglu, A., and Sonmez, T. 1998. Random serial dictatorship and the core from random endowments in house allocation problems. *Econometrica* 66(3):689–701.

Budish, E., and Cantillon, E. 2012. The multi-unit assignment problem: Theory and evidence from course allocation at harvard. *American Economic Review* 102(5):2237–2271.

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

Budish, E.; Othman, A.; and Sandholm, T. 2010. Finding approximate competitive equilibria: Efficient and fair course allocation. *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Budish, E. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy* 119(6):1061–1103.

Goerigk, M.; Hoshino, R.; Kawarabayashi, K.; and Westphal, S. 2014. Solving the traveling tournament problem by packing three-vertex paths. *Proceedings of the 28th AAAI Conference on Artificial Intelligence* (to appear).

Kalinowski, T.; Narodytska, N.; Walsh, T.; and Xia, L. 2013. Strategic behavior when allocating indivisible goods sequentially. *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI'13)*.

Kendall, G.; Knust, S.; Ribeiro, C.; and Urrutia, S. 2010. Scheduling in sports: An annotated bibliography. *Computers and Operations Research* 37:1–19.

Kominers, S.; Ruberry, M.; and Ullman, J. 2010. Course allocation by proxy auction. *Proceedings of the 6th International Conference on Internet and Network Economics (WINE'10)* 551–558.

Krishna, A., and Unver, U. 2008. Improving the efficiency of course bidding at business schools: Field and laboratory studies. *Marketing Science* 27(2):262–282.

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

Roth, A. 2002. The economist as engineer: Game theory, experimentation, and computation as tools for design economics. *Econometrica* 70(4):1341–1378.

Sonmez, T., and Unver, U. 2010. Course bidding at business schools. *International Economic Review* 51(1):99–123.