A Exploration and Extension of the Course Allocation problem

John Wu
Brown University
john_w_wu@brown.edu

Abstract

Course allocation in universities is a challenging combinatorial market-design problem: students have complex, bundle-based preferences over indivisible course seats, and monetary transfers are prohibited. The Approximate Competitive Equilibrium from Equal Incomes (A-CEEI) framework offers strong fairness and efficiency guarantees, but its practical implementation typically limits students to a one-shot preference report and a single reallocation "boost" round. In this work, we attempt to extend A-CEEI to a dynamic, shopping-period setting by isolating that boost step and iterating it under a fixed price vector. We also introduce a lightweight, property-based reporting language that empowers students to express quotas, cross-category adjustments, and soft constraints without prohibitive input costs. Through experiments in small toy instances, we show that our iterative boost can recover additional welfare and inherently discourages strategic misreports via lock-in effects, but is prone to "deadlock" once students exhaust their locked seats. We discuss how richer benchmark generators, standardized fairness metrics, and periodic price or seat adjustments can overcome these limitations, and we outline a path toward a fully dynamic, incentive-compatible course-matching system.

1 Introduction

Assigning students to courses is a foundational yet difficult problem at nearly every university. Each student has individual tastes (e.g. loving an AI seminar but disliking medieval poetry), interaction effects (complements and substitutes among course pairs), and higher-level constraints (caps on electives, balance between STEM and humanities, or social grouping with friends). Traditional centralized matching mechanisms struggle to capture this rich structure without overwhelming students or sacrificing fairness.

The A-CEEI mechanism (Approximate Competitive Equilibrium from Equal Incomes) addresses many of these challenges by giving each student an artificial budget, solving market-clearing prices via a convex program, and then rounding to an integer assignment. While A-CEEI guarantees approximate envy-freeness and high ex-ante efficiency, its real-world deployments typically require students to submit preferences only once and include a single "boost" reallocation that absorbs leftover capacity. As a result, students cannot update their preferences over a real shopping period when new information (syllabus details, peer recommendations) arrives.

In this thesis, we make two contributions. First, we propose a property-based reporting language that lets students encode per-course values, property adjustments (e.g. "at most two AI electives"), and soft quotas without exponential input cost. Second, we isolate the boost step of A-CEEI and iterate it under fixed prices to model a multi-round shopping period. Our experiments on toy examples demonstrate initial welfare gains and strong practical incentive compatibility, but also reveal deadlock phenomena once seats are locked. We conclude by outlining future directions: richer benchmarks, standardized

Preprint. Under review.

fairness metrics, and lightweight price or seat updates to build a truly dynamic, strategy-proof course-matching system.

1.1 Model

The Course Match problem Othman et al. [2010] is fundamentally a combinatorial assignment problem where we want to allocate a set of indivisible items (Courses) across a set of agents (Students), where each agent has a demand over bundles of items (Schedules). Notably, in these problems monetary transfer is prohibited (as is the case in real life).

Mathematically, this problem is thus defined as an tuple

$$\left(S, C, (q_j)_{j=1}^{M}, (\Psi_i)_{i=1}^{N}, (\succsim_i)_{i=1}^{N}\right)$$

Where

1. S is the set of students s.t. $|S| = N$
2. C is the set of courses s.t. $|C| = M$ and there are c_j seats in course j. Allocations must be feasible so $\sum_i x_{ij} \leq q_j \forall j$
3. Ψ_i is the set of permissible schedules for student i. A schedule is a subset of C, i.e., $x_i \subseteq C$, or equivalently a vector $x_i \in \{0,1\}^M$. Each student i has a set of acceptable schedules $\Psi_i \subseteq 2^C$.
4. $\succsim_i$ is a complete, reflexive, and transitive preference relation over Ψ_i. Preferences are assumed to be strict, meaning that $x \succsim_i y$ implies $x \neq y$.

For the Course Match Problem specifically however we can make a few further adjustments:

1. **Loosened Feasibility** - we also introduce q'_j which is a smaller course cap that would be listed on CAB and q_j as the hard cap including overflow seats/overrides. While this isn't relevant to the constraints mentioned earlier it may be relevant to the objective later.
2. **Permissible Schedules** - Permissible schedules are based mainly on the following constraints:
   (a) Student's can't exceed their course capacity k and they must exceed their minimum course requirement k'_i

   $$\sum_{j \in C} x_{ij} \leq k_i, \sum_{j \in C} x_{ij} \geq k'_i, \forall i \in S$$

   (b) Courses can't have overlapping time slots, prerequisites, etc... i.e. if courses j and j' overlap or are prerequisites of each other, or for any other reason cannot be allocated together.

   $$x_{ij} + x'_{ij} \leq 1$$
3. **Strict Preference**: Some algorithms like A-CEEI make the assumption of $\succ_i$ instead of $\succsim_i$ which is fine because we can conrtrol the reporting language and thus the student's expressive power over preferences.
4. **Preference Representation**: There is a natural push and pull between the granularity of $\succ_i$ over Ψ_i and the conciceness and ease of use of the reporting language. Thus in this project we explore a couple of approximations over preferences including pairwise adjustments Othman et al. [2010].

1.2 ACEEI in Practice

Translating the A-CEEI model into a deployable course-matching system entails three main components: a *price-vector search* (tatonnement) to find approximately market-clearing prices, an *integer rounding* step to produce actual schedules, and a one-shot *boost round* to absorb leftover capacity. We draw heavily on Budish [2011] for the first component and on Othman et al. [2010] and Budish et al. [2017a] for the latter stages.

1. Price discovery via tatonnement and price-vector search Starting from an initial price vector $p^{(0)} \geq 0$, the mechanism iterates:

$$p_j^{(t+1)} \;=\; p_j^{(t)} \;+\; \alpha^{(t)}\Big(D_j(p^{(t)}) - C_j\Big) \quad \forall j \in C,$$

where:

- $D_j(p)$ is the *aggregate fractional demand* for course j at prices p, computed by letting each student spend her unit budget to maximize $U_i(S) - \sum_{k \in S} p_k$ over the relaxed schedule set Ψ_i.
- C_j is the course capacity (or the "soft" cap in the CAB listing).
- $\alpha^{(t)}$ is a diminishing step size (e.g. $\alpha^{(t)} \propto 1/\sqrt{t}$) chosen to ensure convergence in the convex relaxation.

This tatonnement process drives excess demand to zero, approximating a competitive equilibrium under divisible goods. Budish [2011] prove that, under mild conditions on $\alpha^{(t)}$, the price vector converges to within ε of market-clearing prices in $O(1/\varepsilon^2)$ iterations. In practice, a *global price-vector search*—often implemented via simulated annealing or coordinate descent—helps escape local oscillations and finds a more envy-free solution when budgets are equalized across students.

2. Integer rounding via mixed integer programming Once prices p^* are fixed, each student i solves the discrete affordability problem:

$$\max_{S \in \Psi_i} \; U_i(S) - \sum_{j \in S} p_j^* \quad \text{s.t.} \; \sum_{j \in S} p_j^* \leq 1,$$

which can be encoded as a small mixed-integer program (MIP) across all students simultaneously, respecting hard caps q_j, time-conflict constraints, and individual load bounds. The solution is an integral allocation that approximates the fractional equilibrium, at the cost of slight envy (up to one course) and reduced efficiency due to rounding.

3. Single boost round for undersubscribed courses After rounding, some courses may remain underfilled. Following Budish et al. [2017a], CM performs one final "boost":

1. Identify all undersubscribed courses j with remaining capacity.
2. Increase each student's budget by a small amount $\delta > 0$, making previously unaffordable courses reachable.
3. Allow each student a *one–time choice* between her current bundle and any bundle that adds an undersubscribed course.

This single pass nudges the allocation toward Pareto improvements without sacrificing the approximate envy-freeness guarantees of A-CEEI.

Remarks and limitations

- The tatonnement step inherits *approximate envy-freeness* (up to one course) and *approximate efficiency* in the fractional domain, as shown in Budish [2011].
- Integer rounding may reintroduce envy and inefficiency, but the boost round limits these distortions.
- The entire process runs *once*; it does not adapt to new information or preference updates during a shopping period.

2 Reporting Language

An allocation mechanism must let students articulate rich, idiosyncratic preferences without drowning them in data entry. Course Match (CM) tackles this by restricting the *reporting language*—the primitives a student can use to build a utility function. CM implements the three–component language proposed by Budish2011; large–scale deployments are reported in Budish et al. [2017b].

2.1 Budish's Three–Component Language

i) **Additive part.** Each student i assigns a value $v_{ij} \in [0, 100]$ to every course $j \in C$.

ii) **Setwise (pairwise) adjustments.** Bonuses or penalties $\mu_{ijj'}$ on *pairs* of courses capture simple complementarities and substitutabilities.

iii) **Logical constraints.** "No more than x of these y courses" are modelled as hard feasibility constraints.

The induced utility for a bundle $S \subseteq C$ is

$$U_i^{\mathrm{CM}}(S) = \sum_{j \in S} v_{ij} + \sum_{\substack{(j,j') \in S \times S \\ j < j'}} \mu_{ijj'}. \tag{1}$$

2.2 Observed Preference Motifs

A few informal interviews with Brown undergraduates and alumni reveal five recurring motives:

1. **Idiosyncratic taste.** "I love *AI*, dislike *Medieval Poetry*."
2. **Pairwise interactions.** Complements (*Philosophy of Mind* only with *CogSci*); substitutes (*Operating Systems vs. Deep Learning*).
3. **Property reasoning.** "$\leq$2 AI electives," "avoid 9 a.m." "take *at least* one seminar."
4. **Categorical balance.** STEM/Humanities or Hard/Easy balance.
5. **Social coupling.** Enrol with a friend if feasible.

Motifs 1–2 fit CM's language; 3–5 do not.

2.3 Where Existing Languages Fail

Table 1: Coverage of observed preference motifs vs. reporting effort. Motifs 1–2 = *pairwise course interactions*; Motifs 3–4 = *property / quota reasoning*. ✓= exact support, ○= partial / clunky support, ✗= not supported.

Reporting language	Motifs 1–2	Motifs 3–4	Input cost
Pairwise / setwise	✓	✗	$O(m^2)$
Full bundle revelation	✓	✓	$O(2^m)$
Property–based (this work)	✓	✓	$O(m^2 + \lvert P \rvert^2)$

2.4 Illustrative Break-downs

Example 1(Cap on AI electives).[1]

- *Setup.* Five AI–tagged courses $A_1, \ldots, A_5$ and any number of non-AI courses. The student's *true* utility for a bundle S is
$$u_{\mathrm{true}}(S) = \begin{cases} 10\, x(S) & \text{if } x(S) \leq 2, \\ -20 & \text{if } x(S) \geq 3, \end{cases}$$
where $x(S) = |S \cap \{A_1, \ldots, A_5\}|$.
Intuition: two AI electives are great, the third blows up the schedule because we only need 2 more AI electives to satisfy our requirements.
- *Pairwise language fails.* Give each AI course an additive value $v = 10$. To discourage the third AI, CM lets us add a *negative* adjustment μ for every *pair* of AI courses chosen. With k AI courses the bundle utility becomes $10k + \mu\binom{k}{2}$. Any choice of $\mu < 0$ that makes $k = 3$

[1] Numerical values are for illustration; any monotone "step–down" utility exhibits the same behavior.

unattractive will already wipe out the benefit of $k = 2$ Preventing *triples* without hurting *pairs* would require negative bonuses on *triples*, *quadruples*, etc.— parameters the pairwise language cannot express. This is possible if we use setwise adjustments like proposed in the original CM paper, but it would be wildly unwieldy and would fail if we have preferences for 2 or 4 AI courses but not 3.

- *Bundle language.* Gives the correct utilities by enumerating all $2^5 = 32$ AI-subsets, but at exponential cost.
- *Property language (ours).* Tag each course with property AI. Specify a single penalty $\gamma^-_{i,\text{AI},2} = -30$, applied once $x(S)$ exceeds 2.

This same logic applies and is extendible to trying to any attempt to impose soft structure on preferences like rewarding taking interesting hard courses but penalizing too many hard courses that would overburden the student or wanting to balance between stem and humanity courses.

2.5 A Property–Based Extension

We propose augmenting each course with a *set of binary properties* $P_j \subseteq P$ (e.g. AI, MORNING, SEMINAR, HARD). The registrar supplies a default set of properties and students may add their own flags ("FRIEND_PLANNING_ON_TAKING"), making the scheme extendable.

Each student reports three simple objects:

i) **Per-course values** v_{ij} (unchanged).

ii) **Property adjustments** $\{\alpha_{ip}\}_{p\in P}$ (bonus/penalty for *each* occurrence of property p) and $\{\beta_{ipp'}\}_{p<p'}$ (interaction of two properties, e.g. HARD with EASY).

iii) **Soft cardinality constraints** $\{\gamma^+_{ip\ell}, \gamma^-_{ip\ell}\}$ giving a bonus (or penalty) whenever the bundle contains at least / more than ℓ courses with property p.

Utility. For a bundle S let $c_p(S) = |\{j \in S : p \in P_j\}|$.

$$
\begin{aligned}
U_i^{\text{prop}}(S) = & \sum_{j\in S} v_{ij} && \text{(idiosyncratic)}\\
& + \sum_{p\in P} \alpha_{ip}\, c_p(S) && \text{(per-property)}\\
& + \sum_{p<p'} \beta_{ipp'}\, c_p(S)\, c_{p'}(S) && \text{(prop–prop)}\\
& + \sum_{p,\ell} [\![\, c_p(S) \geq \ell \,]\!]\, \gamma^+_{ip\ell} && \text{(bonus once quota met)}\\
& + \sum_{p,\ell} [\![\, c_p(S) > \ell \,]\!]\, \gamma^-_{ip\ell} && \text{(penalty beyond cap).}
\end{aligned}
$$

Square-brackets denote indicator functions. Input size is $O(|P|^2 + |P|L)$, where L is the maximum quota index a student chooses to specify (usually small).

Example 1 (≤ 2 AI): Set $\gamma^-_{i,\text{AI},2} = -M$.

Example 2 (3 AI, unless seminar present): $\gamma^-_{i,\text{AI},3} = -M$ and $\beta_{i,\text{AI},\text{SEM}} = +M$.

Example 3 (friend coupling): Student adds property FRIENDTAKING; which attaches a flag that boosts courses that both friends rate highly in their preference bundles.

Computation. Because α, β are linear and the γ-terms are piecewise-linear in $c_p(S)$, U_i^{prop} remains concave; A-CEEI's price-finding LP therefore stays convex, and the integral rounding step is unchanged.

Cognitive cost. Most students interact with ≤ 10 salient properties; they specify at most a handful of γ thresholds, yielding tens—not hundreds—of parameters. By contrast, CM's pairwise language already requires $O(m^2)$ fields.

Expressive Power The property language is also easily extensible to incorperate external information (e.g. like critical review scores and sentiment which might play a soft role in shaping user preferences if they have preferences but are undecided in how to apply them directly to courses).

Take-away. The property language eliminates the *missing-bid* inefficiency by letting students cheaply encode caps, minima, and cross-category preferences that CM cannot express—without the combinatorial blow-ups of full bundle revelation.

A note on future complexity. Large-language-model assistants can already draft pairwise or bundle tables from natural-language prompts [Soumalias et al., 2025]. As these tools mature, the marginal cost of entering *many* parameters falls, tipping the balance toward more expressive languages.

Our property framework is deliberately a step in the direction of introducing a little bit of complexity to target useful expressive power so that students who prefer can keep the short form, while power-users (or LLM agents) can layer richer cross-property adjustments without making the computational burden worse for others.

Other easy wins that push towards expressive power are a combination of pairwise adjustments and bundle preferences where if a bundle preference is specified it overwrites the pairwise and individual calculations.

3 Iterative Algorithms

Building on the three–module view of A-CEEI (tatonnement, rounding, boost), we propose a *simplified iterative boost* mechanism tailored to adapting this algorithm to include shopping period dynamics that allow students to change their value over time. To do this, we first simplified ACEEI by **skipping the full tatonnement** and **reusing a linear program to calculate the price vector**, we isolate the boost step and repeat it each time students update their valuations.

3.1 Algorithm: LP → MIP → Iterative Boost

Each round $t = 0, 1, \ldots, T$ proceeds as follows:

1. **Price computation (once, at $t = 0$).** Solve the *welfare relaxation*

$$\max_{\{x_i \in \text{conv}(\Psi_i)\}} \sum_{i \in S} U_i(x_i) \quad \text{s.t.} \quad \sum_i x_{ij} \leq C_j \ \forall j,$$

   where $U_i(x_i)$ is the linear extension of each student's true bundle utility (including additive and pairwise adjustments). Let p_j be the dual variable for $\sum_i x_{ij} \leq C_j$; these prices remain fixed thereafter.

2. **Allocation with price–budget constraint.** For each student i, solve

$$\max_{S \in \Psi_i} \ U_i(S) \quad \text{s.t.} \quad \sum_{j \in S} p_j \leq B_i^{(t)},$$

   where $B_i^{(0)} = 1$ and $B_i^{(t+1)} = B_i^{(t)} + \delta$. This MIP respects course capacities and schedule constraints.

3. **Boost step.** Identify courses with unfilled seats. Increase each student's budget $B_i^{(t+1)} = B_i^{(t)} + \delta$. Restrict choices to their current bundle $S_i^{(t)}$ or any free course, allowing them to drop courses to free budget.

4. **Update.** Set $S_i^{(t+1)}$ to the chosen bundle. Stop when no student changes or $t = T$.

Evaluation Metric: Budget-Weighted Nash Social Welfare At each round we compute

$$\mathrm{NSW}(t) = \sum_{i \in S} B_i^{(t)} \, \log\big(U_i(S_i^{(t)})\big),$$

which balances fairness (via evolving budgets) and total welfare under our fixed prices.

Notes

- Prices derive from the duals of the welfare-maximizing LP with full bundle utility $U_i(x_i)$.
- We do not re-run price discovery after round 0; interactivity comes solely from the boost-only MIP solves.
- Future work will explore dynamic price updates on a more complex test bed and alternative fairness–efficiency metrics.

3.2 Results

In our experiments on a couple of toy instances, the simplified iterative boost mechanism demonstrates the following behavior:

- The initial LP→MIP allocation attains welfare close to the one-shot solution or pushes towards the welfare and converges in relatively few iterations.
- Subsequent boost rounds recover only a small fraction of the remaining welfare gap before stalling.
- When students' valuations are perturbed mid-process—modeling new information—the algorithm frequently "deadlocks:" no further exchanges are possible under fixed prices and locked seats, and social welfare plateaus.
- Throughout all trials, we observed that any attempt to misreport preferences results in strictly worse outcomes for the deviating student, suggesting that the boost-only loop is effectively incentive-compatible in practice.

3.3 Future

We have looked into one possible shopping-period extension of A-CEEI that isolates and iterates the Pareto-improving boost step while holding prices fixed. Our preliminary findings show:

- *Partial welfare recovery*: Boost-only iterations can improve allocations beyond the initial round but are very limited and are prone to deadlocks when values are perturbed.
- *Practical incentive compatibility*: Lock-in effects discourage strategic misreports, as any deviation locks a student into a suboptimal bundle.
- *Sensitivity to evolving preferences*: Mid-process preference shifts exacerbate local optima and limit convergence to the Nash Social Welfare frontier.

These observations point to several new directions for future work:

- **Selective seat unfreezing.** Introduce a mechanism for "unlocking" a small, bounded number of previously fixed course assignments each round. By temporarily freeing these seats, the system can catalyze new beneficial exchanges without fully rolling back past allocations.
- **Systematic budget perturbations.** Investigate structured perturbation schedules for student budgets—e.g. diminishing random shocks or cyclic boosts—to guide the allocation trajectory. Since extreme budgets collapse to the one-shot equilibrium, carefully calibrated perturbations may help navigate around local welfare plateaus while preserving price-constraint realism.

4 Conclusion and Future Work

In this thesis we explored a "shopping-period" extension of A-CEEI by pulling out and repeating its Pareto-improving boost step while keeping prices fixed. Although the very first round usually gets us

close to the best possible total utility, the boost-only loop often stalls—students get locked into their initial bundles and can't trade further. On the bright side, this lock-in effect also means that trying to game the system backfires: any lie about your true preferences leaves you worse off.

Our experiments so far have been small and somewhat inconclusive. Random toy instances rarely show the kinds of conflicts that really stress a multi-round algorithm. For example, consider a tiny "AB–C puzzle":

```
Courses: A (1 seat), B (1 seat), C (2 seats)
Students:
  • Babba wants A and B together (big bonus for the pair),
  • Cabba only cares about A.
```

A one-shot run will give A to Cabba and B to Babba, leaving room in C—but without price updates or seat releases, our boost loop can't move anyone into C and just gets stuck. That simple example already shows why we need richer test cases and smarter ways to "unjam" the process.

Looking ahead, we see three key steps to make real progress:

- **Build better benchmarks.** Create a library of realistic puzzles—drawn from actual enrollment conflicts or from course popularity distributions—so we can see how the algorithm behaves in more lifelike settings.
- **Measure more than just total welfare.** In addition to Nash Social Welfare, we need tools to check envy-freeness, Pareto efficiency, minimum-utility guarantees, and even simulate strategic reporting by agents.
- **Re-plug into full A-CEEI.** Combine the iterative boost with occasional price updates or controlled seat releases. Finding the minimal tweaks needed to break deadlocks is our next challenge.

Even though our initial trials ran into more roadblocks than clear wins, we believe fully dynamic, strategy-proof course allocation is a solvable problem. By investing in richer benchmarks, standard metrics, and fast prototype infrastructure, future work can turn these ideas into a robust system for real shopping-period settings.

References

Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011. doi: 10.1086/664613.

Eric Budish, Gérard P. Cachon, Judd B. Kessler, and Abraham Othman. Course match: A large-scale implementation of approximate competitive equilibrium from equal incomes for combinatorial allocation. *Operations Research*, 65(2):314–336, 2017a. doi: 10.1287/opre.2016.1544.

Eric Budish, Gérard P. Cachon, Judd B. Kessler, and Abraham Othman. Course match: A large-scale implementation of approximate competitive equilibrium from equal incomes for combinatorial allocation. *Operations Research*, 65(2):314–336, 2017b. doi: 10.1287/opre.2016.1544.

Abdallah Othman, Eric Budish, and Tuomas Sandholm. Finding approximate competitive equilibria: Efficient and fair course allocation. *Proceedings of the ACM Conference on Electronic Commerce*, 2010.

Ermis Soumalias, Yanchen Jiang, Kehang Zhu, Michael Curry, Sven Seuken, and David C. Parkes. Llm-powered preference elicitation in combinatorial assignment. *arXiv preprint arXiv:2502.10308*, 2025. URL `https://arxiv.org/abs/2502.10308`.