Networks and Markets — Lecture 5: Centralized Matching

Nikhil Garg, Cornell Tech

1 Matching markets and unraveling

1.1 Motivating example

Consider a simple job market:

- Two companies C_1, C_2 and one employee L.
- The employee prefers C_1 to C_2 (so $C_1 \succ_L C_2$).
- Company C_1 starts interviewing on Feb. 15.

What should C_2 do if it wants to hire the employee?

- It might interview earlier (e.g., Jan. 15), and/or
- make an "exploding offer" right before C_1 starts (e.g., Feb. 14).

Now suppose you are the employee and you receive an offer from C_2 on Feb. 13. What should you do? If you prefer C_1, you might still feel pressure to accept early, because waiting risks ending up with nothing.

This dynamic can lead to *unraveling*: offers and decisions move earlier and earlier in time.

1.2 Two broad "solutions"

- **Centralized rule/authority.** Someone with power prevents unraveling (e.g., by setting/forcing a common timeline).
- **Centralized matching.** Everyone submits rankings, and a centralized algorithm decides who goes where.

Example: graduate admissions (Council of Graduate Schools). In U.S. graduate admissions, many programs coordinate through the Council of Graduate Schools (CGS) "April 15" resolution: funded offers remain open until a common deadline, and applicants are not required to decide earlier. This kind of coordination reduces "exploding offers" and helps prevent unraveling.

2 Centralized matching: goals and stability

In a centralized matching system:

- Employees submit rankings over companies.
- Companies submit rankings over employees.

- An algorithm produces a matching.

Examples: medical residency matching (doctors to hospitals), school choice (students to schools), and many others.

2.1 What do we want from an algorithm?

Possible goals:

- "Most optimal for society" / many people get top choices,
- strategy-proofness (hard to game),
- fairness,
- **stability**.

2.2 Stability (definition)

A matching is *stable* if there is no *blocking pair*: no employee–company pair (L, C) such that

- L prefers C to the company they are matched with, and
- C prefers L to the employee they are matched with.

Equivalently: there is no pair (employee, company) that both want each other over who the algorithm matched them to.

3 Stable matchings and deferred acceptance

Stable matching is often introduced via the "matching" problem: two sides of a marketplace (e.g., students and schools).

3.1 Boston mechanism (immediate acceptance)

One allocation method is the *Boston mechanism* (also called *immediate acceptance*). In round k, each currently-unassigned student applies to their kth-choice school, and each school *permanently* accepts applicants up to its remaining capacity.

Example run (3 students, 3 schools) for Boston mechanism Consider students $\{1, 2, 3\}$ and schools $\{A, B, C\}$, where each school has one seat.

Preferences

student	ranking	school	ranking
1	$A \succ B \succ C$	A	$1 \succ 2 \succ 3$
2	$A \succ B \succ C$	B	$2 \succ 3 \succ 1$
3	$B \succ A \succ C$	C	$1 \succ 2 \succ 3$

1. **Round 1.** Students 1 and 2 apply to A, student 3 applies to B. School A accepts 1 and rejects 2; school B accepts 3.
2. **Round 2.** Student 2 applies to B, but B is full, so 2 is rejected.
3. **Round 3.** Student 2 applies to C and is accepted.

The outcome is $(1, A)$, $(3, B)$, $(2, C)$.

Not stable. Student 2 prefers B over C, and school B prefers student 2 over its assigned student 3. So $(2, B)$ is a blocking pair.

Not strategy-proof. Student 2 can benefit by misreporting: if student 2 submits $B \succ A \succ C$ instead, then in round 1 both students 2 and 3 apply to B, and B accepts student 2 (its top choice). Student 2 ends up matched to B instead of C.

3.2 Deferred acceptance (DA) algorithm

Algorithm (student-proposing version). Repeat until a stopping condition holds:

- Loop through students who are not currently matched (in some order).
- A student applies to their favorite school that has not rejected them yet.
- A school considers the new applicant:
  - If the new applicant is more preferred than the school's current tentative match, the school rejects its current match and tentatively accepts the new student.
  - Otherwise, the school rejects the new applicant.

Stopping conditions.

- All students are matched; or
- any remaining unmatched students have been rejected by every school.

3.2.1 Example run (3 students, 3 schools)

Consider students $\{1, 2, 3\}$ and schools $\{A, B, C\}$, where each school has one seat.

Preferences. (same as above)

student	ranking	school	ranking
1	$A \succ B \succ C$	A	$1 \succ 2 \succ 3$
2	$A \succ B \succ C$	B	$2 \succ 3 \succ 1$
3	$B \succ A \succ C$	C	$1 \succ 2 \succ 3$

Student-proposing DA.

1. **Round 1.** Process unmatched students sequentially (say in order 1, 2, 3).
   - Student 1 applies to A; A tentatively holds 1.
   - Student 2 applies to A; A prefers its current match (student 1), so it rejects 2.
   - Student 3 applies to B; B tentatively holds 3.
2. **Round 2.** Student 2 is unmatched, so they apply to B; B prefers 2, so it rejects 3 and tentatively holds 2.
3. **Round 3.** Student 3 is now unmatched, so they apply to A; A prefers its current match (student 1), so it rejects 3.
4. **Round 4.** Student 3 applies to C, which tentatively accepts them.

Final matching: $(1, A)$, $(2, B)$, $(3, C)$.

Claim. There is an algorithm called *deferred acceptance* with the following properties:

1. it produces a stable matching;
2. in the student-proposing version, it is strategy-proof for students (truth-telling is a dominant strategy);
3. among all stable matchings, it returns the matching that is "best" for the same side for which it is strategy-proof.

Stability (in this example). Student 2 would prefer A to B, but school A prefers its match (student 1) over student 2. Student 3 would prefer B (or A) to C, but B prefers its match (student 2) over student 3 and A prefers its match (student 1) over student 3. Hence there is no blocking pair, so the matching is stable.

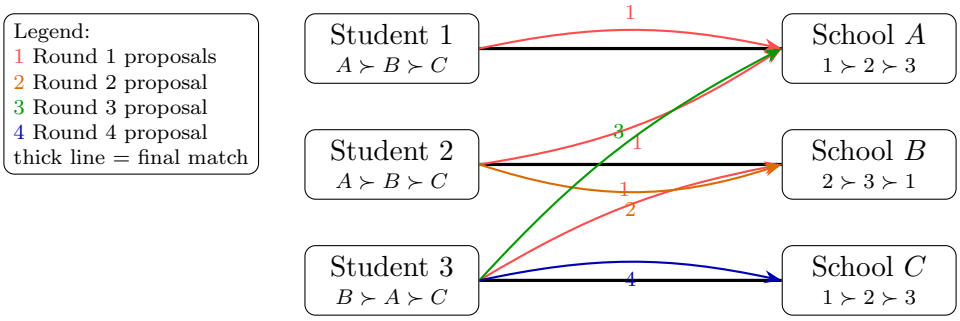

Figure 1: An example run of student-proposing deferred acceptance.

3.2.2 A note on real-world constraints

- It can be important to handle special constraints like couples.
- "Impossible to separate the match from the politics."

Example: prioritizing rural hospitals? Can we "prioritize" rural hospitals? Not in the way one might hope—in every stable matching, the same spots ("rural hospitals") remain unfilled.