Lecture 4: School choice

Thilo Klein

ZEW Mannheim

Introduction

School choice is referred in the literature on market design/matching as giving parents a say in the choice of the schools their children will attend.

In some cities or countries parents have no influence in the selection of the school their children will attend (except by choosing where they live).

But in many cities school districts parents can express preferences about the schools.

School choice is another major application of matching/assignment theory.

Introduction

School choice is referred in the literature on market design/matching as giving parents a say in the choice of the schools their children will attend.

In some cities or countries parents have no influence in the selection of the school their children will attend (except by choosing where they live).

But in many cities school districts parents can express preferences about the schools.

School choice is another major application of matching/assignment theory.

Introduction

School choice is referred in the literature on market design/matching as giving parents a say in the choice of the schools their children will attend.

In some cities or countries parents have no influence in the selection of the school their children will attend (except by choosing where they live).

But in many cities school districts parents can express preferences about the schools.

School choice is another major application of matching/assignment theory.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

The many-to-one assignment model

A school choice model is very close to the many-to-one matching model (e.g., the medical match). There are however some important differences.

A school choice problem is given by:

- A set of students, $I = \{i_1, \ldots, i_n\}$.
- A set of schools, $S = \{s_1, \ldots, s_m\}$.
- For each school $s \in S$ a capacity, q_s, which specifies, for each school, the maximum number of students the school can enroll.
- Each student $i \in I$ has a strict preference ordering P_i over the schools and the option to be unassigned.
- Each school $s \in S$ has a strict priority ordering π_s over the students.

An assignment problem more than a matching problem

The standard case considers public schools. So schools are mere objects and therefore they do not have any preferences.

This is why we assume that schools' priorities rank all students.

Schools' priorities are also assumed to be responsive.

In contrast, students may not find all schools acceptable. Being unassigned can be viewed as:

- home schooling;
- attending a private school.

An assignment problem more than a matching problem

The standard case considers public schools. So schools are mere objects and therefore they do not have any preferences.

This is why we assume that schools' priorities rank all students.

Schools' priorities are also assumed to be responsive.

In contrast, students may not find all schools acceptable. Being unassigned can be viewed as:

- home schooling;
- attending a private school.

An assignment problem more than a matching problem

The standard case considers public schools. So schools are mere objects and therefore they do not have any preferences.

This is why we assume that schools' priorities rank all students.

Schools' priorities are also assumed to be responsive.

In contrast, students may not find all schools acceptable. Being unassigned can be viewed as:

- home schooling;
- attending a private school.

Assignments

Definition

An assignment is a mapping $\mu : I \cup S \to I \cup S$ such that,

- $\mu(i) \in S \cup \{i\}$.

  Each student must be assigned to a school or to himself (the outside option).

- $\mu(s) \subseteq I$.

  Each school is assigned to a subset of students.

- $\mu(i) = s$ if and only if $i \in \mu(s)$.

- $|\mu(s)| \leq q_s$.

Assignments

Definition

An assignment is a mapping $\mu : I \cup S \to I \cup S$ such that,

- $\mu(i) \in S \cup \{i\}$.

  Each student must be assigned to a school or to himself (the outside option).

- $\mu(s) \subseteq I$.

  Each school is assigned to a subset of students.

- $\mu(i) = s$ if and only if $i \in \mu(s)$.

- $|\mu(s)| \leq q_s$.

Assignments

Definition

An assignment is a mapping $\mu : I \cup S \to I \cup S$ such that,

- $\mu(i) \in S \cup \{i\}$.

  Each student must be assigned to a school or to himself (the outside option).

- $\mu(s) \subseteq I$.

  Each school is assigned to a subset of students.

- $\mu(i) = s$ if and only if $i \in \mu(s)$.

- $|\mu(s)| \leq q_s$.

Assignments

Definition

An assignment is a mapping $\mu : I \cup S \to I \cup S$ such that,

- $\mu(i) \in S \cup \{i\}$.

  Each student must be assigned to a school or to himself (the outside option).

- $\mu(s) \subseteq I$.

  Each school is assigned to a subset of students.

- $\mu(i) = s$ if and only if $i \in \mu(s)$.

- $|\mu(s)| \leq q_s$.

Assignments

Definition

An assignment is a mapping $\mu : I \cup S \to I \cup S$ such that,

- $\mu(i) \in S \cup \{i\}$.

  Each student must be assigned to a school or to himself (the outside option).

- $\mu(s) \subseteq I$.

  Each school is assigned to a subset of students.

- $\mu(i) = s$ if and only if $i \in \mu(s)$.

- $|\mu(s)| \leq q_s$.

Assignments

Definition

An assignment is a mapping $\mu : I \cup S \to I \cup S$ such that,

- $\mu(i) \in S \cup \{i\}$.

  Each student must be assigned to a school or to himself (the outside option).

- $\mu(s) \subseteq I$.

  Each school is assigned to a subset of students.

- $\mu(i) = s$ if and only if $i \in \mu(s)$.

- $|\mu(s)| \leq q_s$.

Stability

The concept of stability for school choice problems is similar to the one we used for the medical match.

Definition
An assignment is stable if

- it is individually rational: for each student $i \in I$, $\mu(i)$ is weakly preferred to the option of being unassigned.
- it is non wasteful: for each student $i \in I$,

$$sP_i\mu(i) \qquad \Rightarrow \qquad |\mu(s)| = q_s$$

- there is no **justified envy**: if a student i prefers a school s to his assignment then all students matched to school s must have a higher priority than student i:

$$\text{If } i, j \in I \text{ with } \mu(j) = s \in S \text{ and } sP_i\mu(i) \qquad \Rightarrow \qquad j\pi_s i$$

Stability

The concept of stability for school choice problems is similar to the one we used for the medical match.

Definition

An assignment is stable if

- it is individually rational: for each student $i \in I$, $\mu(i)$ is weakly preferred to the option of being unassigned.
- it is non wasteful: for each student $i \in I$,

$$sP_i\mu(i) \qquad \Rightarrow \qquad |\mu(s)| = q_s$$

- there is no **justified envy**: if a student i prefers a school s to his assignment then all students matched to school s must have a higher priority than student i:

$$\text{If } i, j \in I \text{ with } \mu(j) = s \in S \text{ and } sP_i\mu(i) \qquad \Rightarrow \qquad j\pi_s i$$

Stability

The concept of stability for school choice problems is similar to the one we used for the medical match.

Definition

An assignment is stable if

- it is individually rational: for each student $i \in I$, $\mu(i)$ is weakly preferred to the option of being unassigned.
- it is non wasteful: for each student $i \in I$,

$$sP_i\mu(i) \qquad \Rightarrow \qquad |\mu(s)| = q_s$$

- there is no **justified envy**: if a student i prefers a school s to his assignment then all students matched to school s must have a higher priority than student i:

$$\text{If } i, j \in I \text{ with } \mu(j) = s \in S \text{ and } sP_i\mu(i) \qquad \Rightarrow \qquad j\pi_s i$$

Stability

The concept of stability for school choice problems is similar to the one we used for the medical match.

Definition

An assignment is stable if

- it is individually rational: for each student $i \in I$, $\mu(i)$ is weakly preferred to the option of being unassigned.
- it is non wasteful: for each student $i \in I$,

$$sP_i\mu(i) \qquad \Rightarrow \qquad |\mu(s)| = q_s$$

- there is no **justified envy**: if a student i prefers a school s to his assignment then all students matched to school s must have a higher priority than student i:

$$\text{If } i, j \in I \text{ with } \mu(j) = s \in S \text{ and } sP_i\mu(i) \qquad \Rightarrow \qquad j\pi_s i$$

Stability

The concept of stability for school choice problems is similar to the one we used for the medical match.

Definition

An assignment is stable if

- it is individually rational: for each student $i \in I$, $\mu(i)$ is weakly preferred to the option of being unassigned.
- it is non wasteful: for each student $i \in I$,

$$sP_i\mu(i) \qquad \Rightarrow \qquad |\mu(s)| = q_s$$

- there is no **justified envy**: if a student i prefers a school s to his assignment then all students matched to school s must have a higher priority than student i:

$$\text{If } i,j \in I \text{ with } \mu(j) = s \in S \text{ and } sP_i\mu(i) \qquad \Rightarrow \qquad j\pi_s i$$

Stability

The concept of stability for school choice problems is similar to the one we used for the medical match.

Definition

An assignment is stable if

- it is individually rational: for each student $i \in I$, $\mu(i)$ is weakly preferred to the option of being unassigned.
- it is non wasteful: for each student $i \in I$,

$$sP_i\mu(i) \qquad \Rightarrow \qquad |\mu(s)| = q_s$$

- there is no **justified envy**: if a student i prefers a school s to his assignment then all students matched to school s must have a higher priority than student i:

$$\text{If } i, j \in I \text{ with } \mu(j) = s \in S \text{ and } sP_i\mu(i) \qquad \Rightarrow \qquad j\pi_s i$$

Efficiency

In a school choice problem, since only students have preferences, welfare only takes into account students' preferences.

Definition
An assignment μ is efficient if there is no other assignment μ' such that

- All students *weakly prefer* μ' to μ

  All students are either indifferent between μ' and μ or prefer μ' to μ.

- There is at least one student who strictly prefers μ' to μ

  At least one student who is not assigned to the same school under μ' and μ and prefers the school she is assigned to under μ'.

Efficiency

In a school choice problem, since only students have preferences, welfare only takes into account students' preferences.

Definition

An assignment μ is efficient if there is no other assignment μ' such that

- All students *weakly prefer* μ' to μ

  All students are either indifferent between μ' and μ or prefer μ' to μ.

- There is at least one student who strictly prefers μ' to μ

  At least one student who is not assigned to the same school under μ' and μ and prefers the school she is assigned to under μ'.

Efficiency

In a school choice problem, since only students have preferences, welfare only takes into account students' preferences.

Definition

An assignment μ is efficient if there is no other assignment μ' such that

- All students *weakly prefer* μ' to μ

  All students are either indifferent between μ' and μ or prefer μ' to μ.

- There is at least one student who strictly prefers μ' to μ

  At least one student who is not assigned to the same school under μ' and μ and prefers the school she is assigned to under μ'.

Efficiency

In a school choice problem, since only students have preferences, welfare only takes into account students' preferences.

Definition

An assignment μ is efficient if there is no other assignment μ' such that

- All students *weakly prefer* μ' to μ

  All students are either indifferent between μ' and μ or prefer μ' to μ.

- There is at least one student who strictly prefers μ' to μ

  At least one student who is not assigned to the same school under μ' and μ and prefers the school she is assigned to under μ'.

Efficiency

In a school choice problem, since only students have preferences, welfare only takes into account students' preferences.

Definition

An assignment μ is efficient if there is no other assignment μ' such that

- All students *weakly prefer* μ' to μ

  All students are either indifferent between μ' and μ or prefer μ' to μ.

- There is at least one student who strictly prefers μ' to μ

  At least one student who is not assigned to the same school under μ' and μ and prefers the school she is assigned to under μ'.

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	$\leftarrow$ capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	$\leftarrow$ capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	$\leftarrow$ capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

The assignment μ is not efficient: i_1 and i_3 strictly prefer μ' (i_2 and i_4 are indifferent).

But μ' is efficient.

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	$\leftarrow$ capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

The assignment μ is not efficient: i_1 and i_3 strictly prefer μ' (i_2 and i_4 are indifferent).

But μ' is efficient.

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	$\leftarrow$ capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

The assignment μ is not efficient: i_1 and i_3 strictly prefer μ' (i_2 and i_4 are indifferent).

But μ' is efficient.

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	← capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

The assignment μ' is not stable: (i_2, s_1) is a blocking pair.

But μ is stable.

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	← capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

The assignment μ' is not stable: (i_2, s_1) is a blocking pair.

But μ is stable.

Example

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}
s_2	s_1	s_1	s_2
s_1	s_2	s_2	s_3
s_3	s_3	s_3	s_1

1	2	1	$\leftarrow$ capacity
π_{s_1}	π_{s_2}	π_{s_3}	
i_1	i_3	i_4	
i_2	i_4	i_1	
i_3	i_1	i_2	
i_4	i_2	i_3	

$$\mu = \{(i_1, s_1), (i_2, s_3), (i_3, s_2), (i_4, s_2)\}$$
$$\mu' = \{(i_1, s_2), (i_2, s_3), (i_3, s_1), (i_4, s_2)\}$$

The assignment μ' is not stable: (i_2, s_1) is a blocking pair.

But μ is stable.

Stability v. efficiency

Theorem

It may happen that, for some specific preferences and priorities, a stable assignment is also efficient.

But this is not true in general: it is impossible to guarantee to obtain at the same time efficient and stable assignments.

Stability and efficiency incompatible. But when they coincide, can we select the right matching?

Theorem

There is no efficient and strategy-proof mechanism that selects the efficient and stable matching whenever it exists.

Stability v. efficiency

Theorem

It may happen that, for some specific preferences and priorities, a stable assignment is also efficient.

But this is not true in general: it is impossible to guarantee to obtain at the same time efficient and stable assignments.

Stability and efficiency incompatible. But when they coincide, can we select the right matching?

Theorem

There is no efficient and strategy-proof mechanism that selects the efficient and stable matching whenever it exists.

Stability v. efficiency

Theorem

It may happen that, for some specific preferences and priorities, a stable assignment is also efficient.

But this is not true in general: it is impossible to guarantee to obtain at the same time efficient and stable assignments.

Stability and efficiency incompatible. But when they coincide, can we select the right matching?

Theorem

There is no efficient and strategy-proof mechanism that selects the efficient and stable matching whenever it exists.

Stability v. efficiency

Theorem

It may happen that, for some specific preferences and priorities, a stable assignment is also efficient.

But this is not true in general: it is impossible to guarantee to obtain at the same time efficient and stable assignments.

Stability and efficiency incompatible. But when they coincide, can we select the right matching?

Theorem

There is no efficient and strategy-proof mechanism that selects the efficient and stable matching whenever it exists.

The role of each side of the market

Like for assignment models, we can use algorithms like Deferred Acceptance or Top Trading Cycle (and some new ones).

All those algorithms give a precise role to each side (e.g., proposing for one side, accepting/rejecting for the other side), and two versions of the same algorithm can be obtained, depending on which side is doing what.

For school choice (or assignment problems in general), since objects/schools do not have preferences the attribution of roles is trivial:

- For DA $\rightarrow$ students propose
- For TTC $\rightarrow$ students are assigned what they point to when in cycle.

The role of each side of the market

Like for assignment models, we can use algorithms like Deferred Acceptance or Top Trading Cycle (and some new ones).

All those algorithms give a precise role to each side (e.g., proposing for one side, accepting/rejecting for the other side), and two versions of the same algorithm can be obtained, depending on which side is doing what.

For school choice (or assignment problems in general), since objects/schools do not have preferences the attribution of roles is trivial:

- For DA $\rightarrow$ students propose
- For TTC $\rightarrow$ students are assigned what they point to when in cycle.

The role of each side of the market

Like for assignment models, we can use algorithms like Deferred Acceptance or Top Trading Cycle (and some new ones).

All those algorithms give a precise role to each side (e.g., proposing for one side, accepting/rejecting for the other side), and two versions of the same algorithm can be obtained, depending on which side is doing what.

For school choice (or assignment problems in general), since objects/schools do not have preferences the attribution of roles is trivial:

- For DA $\rightarrow$ students propose
- For TTC $\rightarrow$ students are assigned what they point to when in cycle.

The role of each side of the market

Like for assignment models, we can use algorithms like Deferred Acceptance or Top Trading Cycle (and some new ones).

All those algorithms give a precise role to each side (e.g., proposing for one side, accepting/rejecting for the other side), and two versions of the same algorithm can be obtained, depending on which side is doing what.

For school choice (or assignment problems in general), since objects/schools do not have preferences the attribution of roles is trivial:

- For DA $\rightarrow$ students propose
- For TTC $\rightarrow$ students are assigned what they point to when in cycle.

Deferred Acceptance

The Deferred Acceptance algorithm works like for the medical match:

- Students propose to schools in order of their preferences;
- Schools accept/rejects students' proposals.

The outcome of DA is the student-optimal assignment.

We obtain the usual results:

- DA is strategyproof for the students
- the student-optimal assignment is students' most preferred stable assignment.

Deferred Acceptance

The Deferred Acceptance algorithm works like for the medical match:

- Students propose to schools in order of their preferences;
- Schools accept/rejects students' proposals.

The outcome of DA is the student-optimal assignment.

We obtain the usual results:

- DA is strategyproof for the students
- the student-optimal assignment is students' most preferred stable assignment.

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, i_2, i_3

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, i_2, $\cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, i_2, $\cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
$i_1,\ i_2,\ \cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
$i_1, i_2, \cancel{i_3}$

i_4

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	$\cancel{s_1}$	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, $\cancel{i_2}$, $\cancel{i_3}$

i_4

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	$\cancel{s_1}$	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, $\cancel{i_2}$, $\cancel{i_3}$

i_4

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	$\cancel{s_1}$	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, $\cancel{i_2}$, $\cancel{i_3}$

i_4

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	$\cancel{s_1}$	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, $\cancel{i_2}$, $\cancel{i_3}$
i_4

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	$\cancel{s_1}$	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, $\cancel{i_2}$, $\cancel{i_3}$

i_4

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	$\cancel{s_1}$	$\cancel{s_1}$	$\cancel{s_2}$	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, $\cancel{i_2}$, $\cancel{i_3}$
i_4
i_1, i_4

Immediate acceptance

Another algorithm (popular in practice) is the Immediate Acceptance (IA) algorithm (a.k.a. Boston algorithm).

IA is similar to DA in many aspect:

- students propose to schools in order of the preferences;
- schools accept/reject students

A key difference, however, is that schools acceptance decisions are definitive: once a student is accepted by a school she cannot be rejected at a later step.

Immediate acceptance

Another algorithm (popular in practice) is the Immediate Acceptance (IA) algorithm (a.k.a. Boston algorithm).

IA is similar to DA in many aspect:

- students propose to schools in order of the preferences;
- schools accept/reject students

A key difference, however, is that schools acceptance decisions are definitive: once a student is accepted by a school she cannot be rejected at a later step.

Immediate acceptance

Another algorithm (popular in practice) is the Immediate Acceptance (IA) algorithm (a.k.a. Boston algorithm).

IA is similar to DA in many aspect:

- students propose to schools in order of the preferences;
- schools accept/reject students

A key difference, however, is that schools acceptance decisions are definitive: once a student is accepted by a school she cannot be rejected at a later step.

Immediate acceptance

Another algorithm (popular in practice) is the Immediate Acceptance (IA) algorithm (a.k.a. Boston algorithm).

IA is similar to DA in many aspect:

- students propose to schools in order of the preferences;
- schools accept/reject students

A key difference, however, is that schools acceptance decisions are definitive: once a student is accepted by a school she cannot be rejected at a later step.

The first step of the Immediate Acceptance algorithm is identical to the first step of the Deferred Acceptance algorithm.

- **Step 1**
  Each student applies to her most preferred, acceptable school. (if there is no such school then the student remains unassigned).

  Each school accepts students who propose to it, one by one, following the priority order, up to its capacity. The other students are rejected.

The first step of the Immediate Acceptance algorithm is identical to the first step of the Deferred Acceptance algorithm.

- **Step 1**
  Each student applies to her most preferred, acceptable school. (if there is no such school then the student remains unassigned).

  Each school accepts students who propose to it, one by one, following the priority order, up to its capacity. The other students are rejected.

The first step of the Immediate Acceptance algorithm is identical to the first step of the Deferred Acceptance algorithm.

- **Step 1**
  Each student applies to her most preferred, acceptable school. (if there is no such school then the student remains unassigned).

  Each school accepts students who propose to it, one by one, following the priority order, up to its capacity. The other students are rejected.

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students rejected in the previous step apply to their most preferred, acceptable school among the schools they haven't proposed yet.
(if there is no such school the student remains unassigned).

For each school:

- Students accepted at a previous step remain accepted. The remaining capacity is the school's original capacity minus the number of such students.
- Accepts students who just proposed, up to the remaining capacity following the priority order.
  Remaining students are rejected.

End: The algorithm stops when no student is rejected or all schools have filled their capacities. Any remaining student remains unassigned.

Step $\mathbf{k}, \mathbf{k} \geq \mathbf{2}$

Students rejected in the previous step apply to their most preferred, acceptable school among the schools they haven't proposed yet.
(if there is no such school the student remains unassigned).

For each school:

- Students accepted at a previous step remain accepted. The remaining capacity is the school's original capacity minus the number of such students.
- Accepts students who just proposed, up to the remaining capacity following the priority order.
  Remaining students are rejected.

End: The algorithm stops when no student is rejected or all schools have filled their capacities. Any remaining student remains unassigned.

Step $k, k \geq 2$

Students rejected in the previous step apply to their most preferred, acceptable school among the schools they haven't proposed yet.
(if there is no such school the student remains unassigned).

For each school:

- Students accepted at a previous step remain accepted. The remaining capacity is the school's original capacity minus the number of such students.
- Accepts students who just proposed, up to the remaining capacity following the priority order.
  Remaining students are rejected.

End: The algorithm stops when no student is rejected or all schools have filled their capacities. Any remaining student remains unassigned.

Step $k, k \geq 2$

Students rejected in the previous step apply to their most preferred, acceptable school among the schools they haven't proposed yet.
(if there is no such school the student remains unassigned).

For each school:

- Students accepted at a previous step remain accepted. The remaining capacity is the school's original capacity minus the number of such students.
- Accepts students who just proposed, up to the remaining capacity following the priority order.
  Remaining students are rejected.

End: The algorithm stops when no student is rejected or all schools have filled their capacities. Any remaining student remains unassigned.

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students rejected in the previous step apply to their most preferred, acceptable school among the schools they haven't proposed yet.
(if there is no such school the student remains unassigned).

For each school:

- Students accepted at a previous step remain accepted. The remaining capacity is the school's original capacity minus the number of such students.
- Accepts students who just proposed, up to the remaining capacity following the priority order.
  Remaining students are rejected.

End: The algorithm stops when no student is rejected or all schools have filled their capacities. Any remaining student remains unassigned.

Step $\mathbf{k}, \mathbf{k} \geq \mathbf{2}$

Students rejected in the previous step apply to their most preferred, acceptable school among the schools they haven't proposed yet.
(if there is no such school the student remains unassigned).

For each school:

- Students accepted at a previous step remain accepted. The remaining capacity is the school's original capacity minus the number of such students.
- Accepts students who just proposed, up to the remaining capacity following the priority order.
  Remaining students are rejected.

End: The algorithm stops when no student is rejected or all schools have filled their capacities. Any remaining student remains unassigned.

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, i_2, i_3

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
$i_1,\ i_2,\ \cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, i_2, $\cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
$i_1,\ i_2,\ \cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
$i_1,\ i_2,\ \cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
i_1, i_2, $\cancel{i_3}$

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	$\cancel{s_1}$	s_2	s_2
s_2	s_2	$\cancel{s_2}$	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

Schools	P_{s_1}	P_{s_2}	P_{s_3}
cap.	2	2	1
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

s_1
$i_1,\ i_2,\ \cancel{i_3}$
$i_1,\ i_2$

Top Trading Cycle algorithm

Step 0

For each school $s \in S$, let the remaining capacity be $q_s^1 = q_s$.

Step 1

Students point to their most preferred, acceptable schools(if there is none the student points to herself).

Schools point to the student with the highest priority.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_2^2 = \begin{cases} q_s^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_s^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

Top Trading Cycle algorithm

Step 0

For each school $s \in S$, let the remaining capacity be $q_s^1 = q_s$.

Step 1

Students point to their most preferred, acceptable schools(if there is none the student points to herself).

Schools point to the student with the highest priority.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_2^2 = \begin{cases} q_s^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_s^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

Top Trading Cycle algorithm

Step 0

For each school $s \in S$, let the remaining capacity be $q_s^1 = q_s$.

Step 1

Students point to their most preferred, acceptable schools(if there is none the student points to herself).

Schools point to the student with the highest priority.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_2^2 = \begin{cases} q_s^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_s^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

Top Trading Cycle algorithm

Step 0

For each school $s \in S$, let the remaining capacity be $q_s^1 = q_s$.

Step 1

Students point to their most preferred, acceptable schools(if there is none the student points to herself).

Schools point to the student with the highest priority.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_2^2 = \begin{cases} q_s^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_s^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

Top Trading Cycle algorithm

Step 0

For each school $s \in S$, let the remaining capacity be $q_s^1 = q_s$.

Step 1

Students point to their most preferred, acceptable schools(if there is none the student points to herself).

Schools point to the student with the highest priority.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_2^2 = \begin{cases} q_s^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_s^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

Top Trading Cycle algorithm

Step 0

For each school $s \in S$, let the remaining capacity be $q_s^1 = q_s$.

Step 1

Students point to their most preferred, acceptable schools(if there is none the student points to herself).

Schools point to the student with the highest priority.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_2^2 = \begin{cases} q_s^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_s^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students point to their most preferred, acceptable school whose remaining capacity is not zero (if there is none the student points to herself).

Schools point to the student with the highest priority among the students still present in the problem.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q^2_{k+1} = \begin{cases} q^1_k - 1 & \text{if } s \text{ is in a cycle} \\ q^1_k & \text{if } s \text{ is in not a cycle} \end{cases}$$

End

The algorithm stops when all students or all schools have been removed. Any remaining student is assigned to herself.

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students point to their most preferred, acceptable school whose remaining capacity is not zero (if there is none the student points to herself).

Schools point to the student with the highest priority among the students still present in the problem.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_{k+1}^{2} = \begin{cases} q_k^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_k^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

End

The algorithm stops when all students or all schools have been removed. Any remaining student is assigned to herself.

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students point to their most preferred, acceptable school whose remaining capacity is not zero (if there is none the student points to herself).

Schools point to the student with the highest priority among the students still present in the problem.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q^2_{k+1} = \begin{cases} q^1_k - 1 & \text{if } s \text{ is in a cycle} \\ q^1_k & \text{if } s \text{ is in not a cycle} \end{cases}$$

End

The algorithm stops when all students or all schools have been removed. Any remaining student is assigned to herself.

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students point to their most preferred, acceptable school whose remaining capacity is not zero (if there is none the student points to herself).

Schools point to the student with the highest priority among the students still present in the problem.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_{k+1}^2 = \begin{cases} q_k^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_k^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

End

The algorithm stops when all students or all schools have been removed. Any remaining student is assigned to herself.

Step $\mathbf{k}, \mathbf{k} \geq 2$

Students point to their most preferred, acceptable school whose remaining capacity is not zero (if there is none the student points to herself).

Schools point to the student with the highest priority among the students still present in the problem.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q_{k+1}^2 = \begin{cases} q_k^1 - 1 & \text{if } s \text{ is in a cycle} \\ q_k^1 & \text{if } s \text{ is in not a cycle} \end{cases}$$

End

The algorithm stops when all students or all schools have been removed. Any remaining student is assigned to herself.

Step $\mathbf{k}, \mathbf{k} \geq \mathbf{2}$

Students point to their most preferred, acceptable school whose remaining capacity is not zero (if there is none the student points to herself).

Schools point to the student with the highest priority among the students still present in the problem.

A student in a cycle is assigned the school she is pointing to (or unassigned if pointing to herself) and is removed from the problem .

$$q^2_{k+1} = \begin{cases} q^1_k - 1 & \text{if } s \text{ is in a cycle} \\ q^1_k & \text{if } s \text{ is in not a cycle} \end{cases}$$

End

The algorithm stops when all students or all schools have been removed. Any remaining student is assigned to herself.

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_1	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

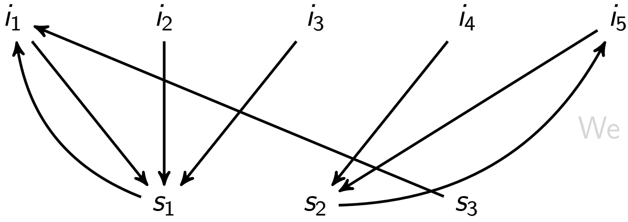
Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_1	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5



We have 2 cycles:

i_1 gets s_1

i_5 gets s_2

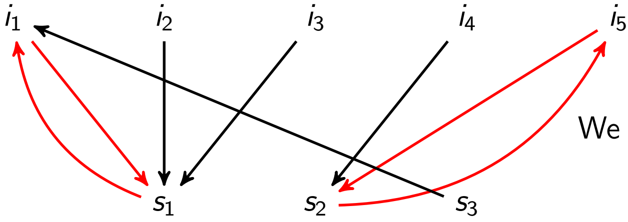
Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_1	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	1	1	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5



We have 2 cycles:

i_1 gets s_1

i_5 gets s_2

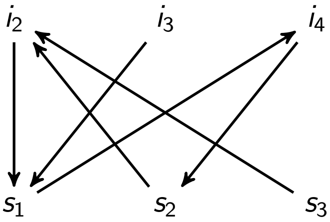
Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_1	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	1	1	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5



We have a cycle:

i_2 gets s_1

i_4 gets s_2

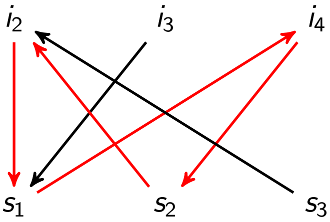
Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	0	0	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5



We have a cycle:

i_2 gets s_1

i_4 gets s_2

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	0	0	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

$$\begin{tikzcd} i_3 \arrow[r, bend left] & s_3 \arrow[l, bend left] \end{tikzcd}$$

We have a cycle:

i_3 gets s_3

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	0	0	0
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

$$\begin{tikzcd} i_3 \arrow[r, bend left] & s_3 \arrow[l, bend left] \end{tikzcd}$$

We have a cycle:

i_3 gets s_3

Example

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	2
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

Final assignment:

$$\mu(i_1) = s_1 \qquad \mu(i_4) = s_2$$
$$\mu(i_2) = s_1 \qquad \mu(i_5) = s_2$$
$$\mu(i_3) = s_3$$

Comparing the assignments

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	1
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

i_1	i_2	i_3	i_4	i_5	stable?	Efficient?

Comparing the assignments

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	1
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

	i_1	i_2	i_3	i_4	i_5	stable?	Efficient?	
DA	s_1	s_2	s_3	s_1	s_2	Yes	No	i_2 & i_4 can swap.

Comparing the assignments

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	1
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

	i_1	i_2	i_3	i_4	i_5	stable?	Efficient?	
DA	s_1	s_2	s_3	s_1	s_2	Yes	No	
IA	s_1	s_1	s_3	s_2	s_2	No	Yes	(i_3, s_2) block.

Comparing the assignments

Students

P_{i_1}	P_{i_2}	P_{i_3}	P_{i_4}	P_{i_5}
s_1	s_1	s_1	s_2	s_2
s_2	s_2	s_2	s_1	s_1
s_3	s_3	s_3	s_3	s_3

Schools

cap.	2	2	1
	P_{s_1}	P_{s_2}	P_{s_3}
	i_1	i_5	i_1
	i_4	i_2	i_2
	i_2	i_3	i_3
	i_3	i_4	i_4
	i_5	i_1	i_5

	i_1	i_2	i_3	i_4	i_5	stable?	Efficient?	
DA	s_1	s_2	s_3	s_1	s_2	Yes	No	
IA	s_1	s_1	s_3	s_2	s_2	No	Yes	
TTC	s_1	s_1	s_3	s_2	s_2	No	Yes	(i_3, s_2) block.

Remark

In general, IA and TTC need not coincide (there might be multiple efficient assignments).

DA and TTC are strategyproof. What about IA?

- i_3 and s_2 block the assignment obtained with IA.
- The problem for i_3 is that by the time she asks s_2 this latter is already full.
- A better strategy for i_3 is to ask in Step 1 of IA school s_2:
  - She would "compete" with i_4 and i_5.
  - Having a higher priority than i_4 she would be accepted.

$\Rightarrow$ IA is not strategyproof.

Remark

In general, IA and TTC need not coincide (there might be multiple efficient assignments).

DA and TTC are strategyproof. What about IA?

- i_3 and s_2 block the assignment obtained with IA.
- The problem for i_3 is that by the time she asks s_2 this latter is already full.
- A better strategy for i_3 is to ask in Step 1 of IA school s_2:
  - She would "compete" with i_4 and i_5.
  - Having a higher priority than i_4 she would be accepted.

$\Rightarrow$ IA is not strategyproof.

Remark

In general, IA and TTC need not coincide (there might be multiple efficient assignments).

DA and TTC are strategyproof. What about IA?

- i_3 and s_2 block the assignment obtained with IA.
- The problem for i_3 is that by the time she asks s_2 this latter is already full.
- A better strategy for i_3 is to ask in Step 1 of IA school s_2:
  - She would "compete" with i_4 and i_5.
  - Having a higher priority than i_4 she would be accepted.

$\Rightarrow$ IA is not strategyproof.

Remark

In general, IA and TTC need not coincide (there might be multiple efficient assignments).

DA and TTC are strategyproof. What about IA?

- i_3 and s_2 block the assignment obtained with IA.
- The problem for i_3 is that by the time she asks s_2 this latter is already full.
- A better strategy for i_3 is to ask in Step 1 of IA school s_2:
  - She would "compete" with i_4 and i_5.
  - Having a higher priority than i_4 she would be accepted.

$\Rightarrow$ IA is not strategyproof.

Remark

In general, IA and TTC need not coincide (there might be multiple efficient assignments).

DA and TTC are strategyproof. What about IA?

- i_3 and s_2 block the assignment obtained with IA.
- The problem for i_3 is that by the time she asks s_2 this latter is already full.
- A better strategy for i_3 is to ask in Step 1 of IA school s_2:
  - She would "compete" with i_4 and i_5.
  - Having a higher priority than i_4 she would be accepted.

$\Rightarrow$ IA is not strategyproof.

Remark

In general, IA and TTC need not coincide (there might be multiple efficient assignments).

DA and TTC are strategyproof. What about IA?

- i_3 and s_2 block the assignment obtained with IA.
- The problem for i_3 is that by the time she asks s_2 this latter is already full.
- A better strategy for i_3 is to ask in Step 1 of IA school s_2:
  - She would "compete" with i_4 and i_5.
  - Having a higher priority than i_4 she would be accepted.

$\Rightarrow$ IA is not strategyproof.

The Immediate Acceptance algorithm is "popular": it is used in many cities.

One of the attractive features for politicians and policy makers is that it maximizes the number of students matched to their top choice.

But economists argue that this argument is flawed:

Parents have to be strategic and put as a first choice a school they believe they will obtain, not the true top choice.

The Immediate Acceptance algorithm is “popular”: it is used in many cities.

One of the attractive features for politicians and policy makers is that it maximizes the number of students matched to their top choice.

But economists argue that this argument is flawed:

Parents have to be strategic and put as a first choice a school they believe they will obtain, not the true top choice.

The Immediate Acceptance algorithm is "popular": it is used in many cities.

One of the attractive features for politicians and policy makers is that it maximizes the number of students matched to their top choice.

But economists argue that this argument is flawed:

Parents have to be strategic and put as a first choice a school they believe they will obtain, not the true top choice.

At a conference organized by the Federal Reserve Bank of Chicago in 1994 ("*Midwest approaches to school reform*"), Meyer and Glazerman report:

> *It may be optimal for some families to be strategic in listing their school choices. For example, if a parent thinks that their favorite school is oversubscribed and they have a close second favorite, they may try to avoid "wasting" their first choice on a very popular school and instead list their number two school first.*

In a meeting of the *West Zone Parents Group* of the city of Boston, it was said

> *One school choice strategy is to find a school you like that is undersubscribed and put it as a top choice, or, find a school that you like that is popular and put it as a first choice and find a school that is less popular for a "safe" second choice.*

At a conference organized by the Federal Reserve Bank of Chicago in 1994 ("*Midwest approaches to school reform*"), Meyer and Glazerman report:

> *It may be optimal for some families to be strategic in listing their school choices. For example, if a parent thinks that their favorite school is oversubscribed and they have a close second favorite, they may try to avoid "wasting" their first choice on a very popular school and instead list their number two school first.*

In a meeting of the *West Zone Parents Group* of the city of Boston, it was said

> *One school choice strategy is to find a school you like that is undersubscribed and put it as a top choice, or, find a school that you like that is popular and put it as a first choice and find a school that is less popular for a "safe" second choice.*

The Boston school match

The study of school choice with assignment mechanisms started with a paper by Abdulkadiroğlu and Sönmez published in 2003.

The publication of their research was spotted by the Boston Globe, highlighting the flaws of the algorithm used in Boston: the Immediate Acceptance algorithm.

Fall 2003: Abdulkadiroğlu, Pathak, Roth and Sönmez were asked to make some recommendation for the Boston Public Schools.

The Boston school match

The study of school choice with assignment mechanisms started with a paper by Abdulkadiroğlu and Sönmez published in 2003.

The publication of their research was spotted by the Boston Globe, highlighting the flaws of the algorithm used in Boston: the Immediate Acceptance algorithm.

Fall 2003: Abdulkadiroğlu, Pathak, Roth and Sönmez were asked to make some recommendation for the Boston Public Schools.

The Boston school match

The study of school choice with assignment mechanisms started with a paper by Abdulkadiroğlu and Sönmez published in 2003.

The publication of their research was spotted by the Boston Globe, highlighting the flaws of the algorithm used in Boston: the Immediate Acceptance algorithm.

Fall 2003: Abdulkadiroğlu, Pathak, Roth and Sönmez were asked to make some recommendation for the Boston Public Schools.

The Boston school match

The study of school choice with assignment mechanisms started with a paper by Abdulkadiroğlu and Sönmez published in 2003.

The publication of their research was spotted by the Boston Globe, highlighting the flaws of the algorithm used in Boston: the Immediate Acceptance algorithm.

Fall 2003: Abdulkadiroğlu, Pathak, Roth and Sönmez were asked to make some recommendation for the Boston Public Schools.

Boston Public Schools (BPS):

- Over 60,000 students K–12.
- Three zones: East, West and North.
- In 2004, about
  - 4800 students entering Kindergarten
  - 4000 entering 1st grade
  - 4300 entering 6th grade
  - 4000 entering 9th grade.

Prior to 2006, the **Boston Public Schools** (PBS) used the Immediate Acceptance algorithm.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools (zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools (zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools (zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools (zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools (zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools

(zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Boston is a perfect example of our school choice model: students' priorities at schools are set by the administration following specific criteria (i.e., they're not school's *preferences*):

Schools's priorities are constructed this way:

1st tier: Students with an older sibling attending the school.

2nd tier: Students living in the walk zone of the schools (zones are defined by the Boston Public Schools).

3rd tier: All the other students.

Then,

- 50% of a schools' seat are prioritized according to the three ties.
- 50% of a schools' seat are not prioritized

Priorities are made strict using a random draw.

Which algorithm to replace IA?

A first criterion is to use a strategyproof mechanism. Doing so *"levels the playing field"*:

> Parents with a good understanding of IA were able to take advantage of it an game the system successfully...
>
> ...at the expense of the other parents.

Then, efficiency or stability? If we want

- efficiency $\rightarrow$ use TTC;
- stability $\rightarrow$ use DA (with students proposing).

Which algorithm to replace IA?

A first criterion is to use a strategyproof mechanism. Doing so "*levels the playing field*":

> Parents with a good understanding of IA were able to take advantage of it an game the system successfully. . .
>
> . . . at the expense of the other parents.

Then, efficiency or stability? If we want

- efficiency $\rightarrow$ use TTC;
- stability $\rightarrow$ use DA (with students proposing).

Which algorithm to replace IA?

A first criterion is to use a strategyproof mechanism. Doing so "*levels the playing field*":

> Parents with a good understanding of IA were able to take advantage of it an game the system successfully...
>
> ...at the expense of the other parents.

Then, efficiency or stability? If we want

- efficiency $\rightarrow$ use TTC;
- stability $\rightarrow$ use DA (with students proposing).

Choosing between DA and TTC depends on the way we interpret schools' priorities:

- TTC implicitly assumes that students can trade their priorities.
- with DA (stability), priorities are not tradable, students have no ownership on their priorities.

At the outset the task force preferred TTC. Eventually, they settled for DA, which Boston started to use in 2007.

Choosing between DA and TTC depends on the way we interpret schools' priorities:

- TTC implicitly assumes that students can trade their priorities.
- with DA (stability), priorities are not tradable, students have no ownership on their priorities.

At the outset the task force preferred TTC. Eventually, they settled for DA, which Boston started to use in 2007.

Choosing between DA and TTC depends on the way we interpret schools' priorities:

- TTC implicitly assumes that students can trade their priorities.
- with DA (stability), priorities are not tradable, students have no ownership on their priorities.

At the outset the task force preferred TTC. Eventually, they settled for DA, which Boston started to use in 2007.

Choosing between DA and TTC depends on the way we interpret schools' priorities:

- TTC implicitly assumes that students can trade their priorities.
- with DA (stability), priorities are not tradable, students have no ownership on their priorities.

At the outset the task force preferred TTC. Eventually, they settled for DA, which Boston started to use in 2007.

The New York City school match

The NYC case is different from Boston in several aspects:

- Much larger scale: 90,000+ students, 500+ different academic programs (high school).
- The source of debate was the National Resident Matching Program: officials at the NYC Department of Education wondered if it could be adapted to NYC.

School match in NYC was initially decentralized (creating waiting lists), and students were restricted about the number of applications they could send.

The New York City school match

The NYC case is different from Boston in several aspects:

- Much larger scale: 90,000+ students, 500+ different academic programs (high school).
- The source of debate was the National Resident Matching Program: officials at the NYC Department of Education wondered if it could be adapted to NYC.

School match in NYC was initially decentralized (creating waiting lists), and students were restricted about the number of applications they could send.

The New York City school match

The NYC case is different from Boston in several aspects:

- Much larger scale: 90,000+ students, 500+ different academic programs (high school).
- The source of debate was the National Resident Matching Program: officials at the NYC Department of Education wondered if it could be adapted to NYC.

School match in NYC was initially decentralized (creating waiting lists), and students were restricted about the number of applications they could send.

The New York City school match

The NYC case is different from Boston in several aspects:

- Much larger scale: 90,000+ students, 500+ different academic programs (high school).
- The source of debate was the National Resident Matching Program: officials at the NYC Department of Education wondered if it could be adapted to NYC.

School match in NYC was initially decentralized (creating waiting lists), and students were restricted about the number of applications they could send.

The New York City school match

The NYC case is different from Boston in several aspects:

- Much larger scale: 90,000+ students, 500+ different academic programs (high school).
- The source of debate was the National Resident Matching Program: officials at the NYC Department of Education wondered if it could be adapted to NYC.

School match in NYC was initially decentralized (creating waiting lists), and students were restricted about the number of applications they could send.

Schools in NYC are not homogeneous:

- Some schools can screen students: targeting students with specific needs and skills

  $\Rightarrow$ these schools have preferences over students.

- Other schools are more “classic”, like in Boston.

Evidence of schools being strategic (in the decentralized procedure) convinced that school choice in NYC is a matching problem and not an assignment problem.

$\Rightarrow$ Deferred Acceptance is the natural choice for NYC.

Schools in NYC are not homogeneous:

- Some schools can screen students: targeting students with specific needs and skills

  $\Rightarrow$ these schools have preferences over students.

- Other schools are more "classic", like in Boston.

Evidence of schools being strategic (in the decentralized procedure) convinced that school choice in NYC is a matching problem and not an assignment problem.

$\Rightarrow$ Deferred Acceptance is the natural choice for NYC.

Schools in NYC are not homogeneous:

- Some schools can screen students: targeting students with specific needs and skills

  $\Rightarrow$ these schools have preferences over students.

- Other schools are more "classic", like in Boston.

Evidence of schools being strategic (in the decentralized procedure) convinced that school choice in NYC is a matching problem and not an assignment problem.

$\Rightarrow$ Deferred Acceptance is the natural choice for NYC.

Schools in NYC are not homogeneous:

- Some schools can screen students: targeting students with specific needs and skills

  ⇒ these schools have preferences over students.

- Other schools are more "classic", like in Boston.

Evidence of schools being strategic (in the decentralized procedure) convinced that school choice in NYC is a matching problem and not an assignment problem.

⇒ Deferred Acceptance is the natural choice for NYC.

Which version of DA?

Since both sides of the market are strategic, we have two options: the student proposing or the school proposing DA.

Choosing the student proposing version quickly appeared to be the best option:

- DA is strategyproof for students with the student proposing. It produces the student-optimal matching
- For many-to-one problems there is no mechanism that is strategyproof for the schools and that produces stable matchings.

  Here strategyproof = revealing true preferences & true capacity.

Which version of DA?

Since both sides of the market are strategic, we have two options: the student proposing or the school proposing DA.

Choosing the student proposing version quickly appeared to be the best option:

- DA is strategyproof for students with the student proposing. It produces the student-optimal matching
- For many-to-one problems there is no mechanism that is strategyproof for the schools and that produces stable matchings.

  Here strategyproof = revealing true preferences & true capacity.

Which version of DA?

Since both sides of the market are strategic, we have two options: the student proposing or the school proposing DA.

Choosing the student proposing version quickly appeared to be the best option:

- DA is strategyproof for students with the student proposing. It produces the student-optimal matching
- For many-to-one problems there is no mechanism that is strategyproof for the schools and that produces stable matchings.

  Here strategyproof = revealing true preferences & true capacity.

Which version of DA?

Since both sides of the market are strategic, we have two options: the student proposing or the school proposing DA.

Choosing the student proposing version quickly appeared to be the best option:

- DA is strategyproof for students with the student proposing. It produces the student-optimal matching
- For many-to-one problems there is no mechanism that is strategyproof for the schools and that produces stable matchings.

  Here strategyproof = revealing true preferences & true capacity.

The first year of operation of DA in NYC:

- 70,000 students matched to a school on their initial choice list (an increase of 20,000 compared to previous years).
- Unmatched students are ask to submit a new preference list.
- At the end, unmatched students are matched administratively (to a school not on their choice list).
  - Previous system: 30,000 students
  - With DA: 3,000 students.

The first year of operation of DA in NYC:

- 70,000 students matched to a school on their initial choice list (an increase of 20,000 compared to previous years).
- Unmatched students are ask to submit a new preference list.
- At the end, unmatched students are matched administratively (to a school not on their choice list).
    - Previous system: 30,000 students
    - With DA: 3,000 students.

The first year of operation of DA in NYC:

- 70,000 students matched to a school on their initial choice list (an increase of 20,000 compared to previous years).
- Unmatched students are ask to submit a new preference list.
- At the end, unmatched students are matched administratively (to a school not on their choice list).
  - Previous system: 30,000 students
  - With DA: 3,000 students.

The first year of operation of DA in NYC:

- 70,000 students matched to a school on their initial choice list (an increase of 20,000 compared to previous years).
- Unmatched students are ask to submit a new preference list.
- At the end, unmatched students are matched administratively (to a school not on their choice list).
    - Previous system: 30,000 students
    - With DA: 3,000 students.

Weak priorities

In the basic school choice model schools' priorities are assumed to be strict. This assumption is difficult to justify.

Priorities are generally set by policy makers, following simple criteria, defining broad categories like:

- students with a sibling in the school
- students living in the "walking zone"
- students from specific socio-economic or ethnic groups

$\Rightarrow$ students in the same tier have the same priority.

Weak priorities

In the basic school choice model schools' priorities are assumed to be strict. This assumption is difficult to justify.

Priorities are generally set by policy makers, following simple criteria, defining broad categories like:

- students with a sibling in the school
- students living in the "walking zone"
- students from specific socio-economic or ethnic groups

$\Rightarrow$ students in the same tier have the same priority.

Weak priorities

In the basic school choice model schools' priorities are assumed to be strict. This assumption is difficult to justify.

Priorities are generally set by policy makers, following simple criteria, defining broad categories like:

- students with a sibling in the school
- students living in the "walking zone"
- students from specific socio-economic or ethnic groups

$\Rightarrow$ students in the same tier have the same priority.

Weak priorities

In the basic school choice model schools' priorities are assumed to be strict. This assumption is difficult to justify.

Priorities are generally set by policy makers, following simple criteria, defining broad categories like:

- students with a sibling in the school
- students living in the "walking zone"
- students from specific socio-economic or ethnic groups

$\Rightarrow$ students in the same tier have the same priority.

An example of a weak priority for a school s:

P_s
Alice, Bob
Carol
Denis, Erin, Fred
Gilda

- Alice and Bob have a higher priority than any other student.

  But Alice (Bob) doesn't have a higher priority than Bob (Alice).
- Carol has
  - lower priority than Alice and Bob
  - higher priority than Denis, Erin, Fred and Gilda.

An example of a weak priority for a school s:

P_s
Alice, Bob
Carol
Denis, Erin, Fred
Gilda

- Alice and Bob have a higher priority than any other student.

  But Alice (Bob) doesn't have a higher priority than Bob (Alice).
- Carol has
  - lower priority than Alice and Bob
  - higher priority than Denis, Erin, Fred and Gilda.

An example of a weak priority for a school s:

P_s
Alice, Bob
Carol
Denis, Erin, Fred
Gilda

- Alice and Bob have a higher priority than any other student.

  But Alice (Bob) doesn't have a higher priority than Bob (Alice).

- Carol has
  - lower priority than Alice and Bob
  - higher priority than Denis, Erin, Fred and Gilda.

To run an algorithm like DA we need strict priorities. Ties can be (for instance) broken randomly.

But doing so may be inefficient.

Before explaining this, we need to take into account that stability is defined with respect to the original priorities (before breaking ties).

$$\bar{\pi}_s = [\text{Alice}, \text{Bob}], \text{Carol}, [\text{Denis}, \text{ Erin}, \text{ Fred}], \text{Gilda}$$

School s has one seat, it's Alice's most preferred school.

- Alice can block if Denis is assigned to s: she has a strictly higher priority.
- Alice cannot block if Bob is assigned to s: she has the same priority as him.

To run an algorithm like DA we need strict priorities. Ties can be (for instance) broken randomly.

But doing so may be inefficient.

Before explaining this, we need to take into account that stability is defined with respect to the original priorities (before breaking ties).

$$\bar{\pi}_s = [\text{Alice}, \text{Bob}], \text{Carol}, [\text{Denis}, \text{ Erin}, \text{ Fred}], \text{Gilda}$$

School s has one seat, it's Alice's most preferred school.

- Alice can block if Denis is assigned to s: she has a strictly higher priority.
- Alice cannot block if Bob is assigned to s: she has the same priority as him.

To run an algorithm like DA we need strict priorities. Ties can be (for instance) broken randomly.

But doing so may be inefficient.

Before explaining this, we need to take into account that stability is defined with respect to the original priorities (before breaking ties).

$$\bar{\pi}_s = [\text{Alice}, \text{Bob}], \text{Carol}, [\text{Denis}, \text{ Erin}, \text{ Fred}], \text{Gilda}$$

School s has one seat, it's Alice's most preferred school.

- Alice can block if Denis is assigned to s: she has a strictly higher priority.
- Alice cannot block if Bob is assigned to s: she has the same priority as him.

To run an algorithm like DA we need strict priorities. Ties can be (for instance) broken randomly.

But doing so may be inefficient.

Before explaining this, we need to take into account that stability is defined with respect to the original priorities (before breaking ties).

$$\bar{\pi}_s = [\text{Alice}, \text{Bob}], \text{Carol}, [\text{Denis}, \text{ Erin}, \text{ Fred}], \text{Gilda}$$

School s has one seat, it's Alice's most preferred school.

- Alice can block if Denis is assigned to s: she has a strictly higher priority.
- Alice cannot block if Bob is assigned to s: she has the same priority as him.

To run an algorithm like DA we need strict priorities. Ties can be (for instance) broken randomly.

But doing so may be inefficient.

Before explaining this, we need to take into account that stability is defined with respect to the original priorities (before breaking ties).

$$\bar{\pi}_s = [\text{Alice}, \text{Bob}], \text{Carol}, [\text{Denis}, \text{ Erin}, \text{ Fred}], \text{Gilda}$$

School s has one seat, it's Alice's most preferred school.

- Alice can block if Denis is assigned to s: she has a strictly higher priority.
- Alice cannot block if Bob is assigned to s: she has the same priority as him.

Efficiency loss

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

Suppose we break ties with the following order:

Alice, Bob, Carol

So for s_1 after breaking ties Bob has a strictly higher priority than Carol

Efficiency loss

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

Suppose we break ties with the following order:

Alice, Bob, Carol

So for s_1 after breaking ties Bob has a strictly higher priority than Carol

Efficiency loss

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

Suppose we break ties with the following order:

Alice, Bob, Carol

So for s_1 after breaking ties Bob has a strictly higher priority than Carol

P_{Alice}	P_{Bob}	P_{Carol}	π_{s_1}	π_{s_2}	π_{s_3}
s_2	s_3	s_2	Alice	Bob	Carol
s_1	s_2	s_3	Bob	Alice	Alice
s_3	s_1	s_1	Carol	Carol	Bob

Running DA (students proposing) we obtain

$$\mu(\text{Alice}) = s_1, \quad \mu(\text{Bob}) = s_2, \qquad \text{and} \qquad \mu(\text{Carol}) = s_3 \ .$$

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

$$\mu(\text{Alice}) = s_1, \quad \mu(\text{Bob}) = s_2, \qquad \text{and} \qquad \mu(\text{Carol}) = s_3 \ .$$

But the following is also a stable matching (preferred by Bob and Carol):

$$\mu'(\text{Alice}) = s_1, \quad \mu'(\text{Bob}) = s_3, \qquad \text{and} \qquad \mu'(\text{Carol}) = s_2 \ .$$

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

$$\mu(\text{Alice}) = s_1, \quad \mu(\text{Bob}) = s_2, \qquad \text{and} \qquad \mu(\text{Carol}) = s_3 \ .$$

But the following is also a stable matching (preferred by Bob and Carol):

$$\mu'(\text{Alice}) = s_1, \quad \mu'(\text{Bob}) = s_3, \qquad \text{and} \qquad \mu'(\text{Carol}) = s_2 \ .$$

We now have two sources of inefficiency:

- Due to the conflict between stability and Pareto efficiency (in general they're not compatible);
- Weak priorities can generate an assignment that is not the student-optimal assignment.

One way to restor efficiency (from the second source) is to allow students trade their enrollements...

...So we can use the Top Trading Cycle algorithm!

But there's a caveat: we may loose stability if the pointing is not defined properly.

We now have two sources of inefficiency:

- Due to the conflict between stability and Pareto efficiency (in general they're not compatible);
- Weak priorities can generate an assignment that is not the student-optimal assignment.

One way to restor efficiency (from the second source) is to allow students trade their enrollements. . .

. . . So we can use the Top Trading Cycle algorithm!

But there's a caveat: we may loose stability if the pointing is not defined properly.

We now have two sources of inefficiency:

- Due to the conflict between stability and Pareto efficiency (in general they're not compatible);
- Weak priorities can generate an assignment that is not the student-optimal assignment.

One way to restor efficiency (from the second source) is to allow students trade their enrollements…

…So we can use the Top Trading Cycle algorithm!

But there's a caveat: we may loose stability if the pointing is not defined properly.

We now have two sources of inefficiency:

- Due to the conflict between stability and Pareto efficiency (in general they're not compatible);
- Weak priorities can generate an assignment that is not the student-optimal assignment.

One way to restor efficiency (from the second source) is to allow students trade their enrollements...

...So we can use the Top Trading Cycle algorithm!

But there's a caveat: we may loose stability if the pointing is not defined properly.

We now have two sources of inefficiency:

- Due to the conflict between stability and Pareto efficiency (in general they're not compatible);
- Weak priorities can generate an assignment that is not the student-optimal assignment.

One way to restor efficiency (from the second source) is to allow students trade their enrollements...

...So we can use the Top Trading Cycle algorithm!

But there's a caveat: we may loose stability if the pointing is not defined properly.

We now have two sources of inefficiency:

- Due to the conflict between stability and Pareto efficiency (in general they're not compatible);
- Weak priorities can generate an assignment that is not the student-optimal assignment.

One way to restor efficiency (from the second source) is to allow students trade their enrollements...

...So we can use the Top Trading Cycle algorithm!

But there's a caveat: we may loose stability if the pointing is not defined properly.

Improvement cycles

The idea (due to Erdil and Ergin) is to restrict the schools to which a student can point.

- We start from an assignment.
- A student i can point to a school s only if:
  - it is preferred to her assignment.
  - Among all students who prefer s to their assignment student i is among the highest priority students.
- If there's a cycle, we perform trade and we start again until no new trades are realized.

Improvement cycles

The idea (due to Erdil and Ergin) is to restrict the schools to which a student can point.

- We start from an assignment.
- A student i can point to a school s only if:
  - it is preferred to her assignment.
  - Among all students who prefer s to their assignment student i is among the highest priority students.
- If there's a cycle, we perform trade and we start again until no new trades are realized.

Improvement cycles

The idea (due to Erdil and Ergin) is to restrict the schools to which a student can point.

- We start from an assignment.
- A student i can point to a school s only if:
  - it is preferred to her assignment.
  - Among all students who prefer s to their assignment student i is among the highest priority students.
- If there's a cycle, we perform trade and we start again until no new trades are realized.

Improvement cycles

The idea (due to Erdil and Ergin) is to restrict the schools to which a student can point.

- We start from an assignment.
- A student i can point to a school s only if:
  - it is preferred to her assignment.
  - Among all students who prefer s to their assignment student i is among the highest priority students.
- If there's a cycle, we perform trade and we start again until no new trades are realized.

Improvement cycles

The idea (due to Erdil and Ergin) is to restrict the schools to which a student can point.

- We start from an assignment.
- A student i can point to a school s only if:
  - it is preferred to her assignment.
  - Among all students who prefer s to their assignment student i is among the highest priority students.
- If there's a cycle, we perform trade and we start again until no new trades are realized.

Improvement cycles

The idea (due to Erdil and Ergin) is to restrict the schools to which a student can point.

- We start from an assignment.
- A student i can point to a school s only if:
  - it is preferred to her assignment.
  - Among all students who prefer s to their assignment student i is among the highest priority students.
- If there's a cycle, we perform trade and we start again until no new trades are realized.

Example

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

Example

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

$$\mu(\text{Alice}) = s_1, \quad \mu(\text{Bob}) = s_2, \qquad \text{and} \qquad \mu(\text{Carol}) = s_3\ .$$

- Alice and Carol both want s_2. They have the same priority, so they can point to Bob (enrolled at s_2).
- Bob is the only one who want s_3, so he points to Carol.

Example

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

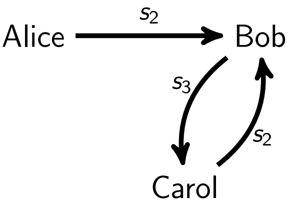
$$\mu(\text{Alice}) = s_1, \quad \mu(\text{Bob}) = s_2, \qquad \text{and} \qquad \mu(\text{Carol}) = s_3 \ .$$

- Alice and Carol both want s_2. They have the same priority, so they can point to Bob (enrolled at s_2).
- Bob is the only one who want s_3, so he points to Carol.

Example

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]



Example

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_3	s_2
s_1	s_2	s_3
s_3	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Bob	Carol
[Bob, Carol]	[Alice, Carol]	[Alice, Bob]

Then we get

$$\mu'(\text{Alice}) = s_1, \quad \mu'(\text{Bob}) = s_3, \qquad \text{and} \qquad \mu'(\text{Carol}) = s_2 \ .$$

No new trades are realized, we stop.

Theorem

If μ is a stable assignment and μ is Pareto dominated by another assignment μ' then there exists a stable improvement cycle.

Using data from NYC, Abdulkadiroğlu, Pathak and Roth find that for the years 2003–2007 they can improve on average the assignment of about 1,700 students (around 2.5% of the students).

Theorem

If μ is a stable assignment and μ is Pareto dominated by another assignment μ' then there exists a stable improvement cycle.

Using data from NYC, Abdulkadiroğlu, Pathak and Roth find that for the years 2003–2007 they can improve on average the assignment of about 1,700 students (around 2.5% of the students).

DA + improvement cycles is not strategyproof.

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_2	s_1
s_3	s_3	s_2
s_1	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Carol	Carol
Bob	[Alice, Bob]	Bob
Carol		Alice

There are two stable assignments:

$$\mu(\text{Alice}) = s_2, \quad \mu(\text{Bob}) = s_3, \quad \text{and} \quad \mu(\text{Carol}) = s_1\ .$$
$$\mu'(\text{Alice}) = s_3, \quad \mu'(\text{Bob}) = s_2, \quad \text{and} \quad \mu'(\text{Carol}) = s_1\ .$$

Let

$$P'_{\text{Alice}} = s_2, s_1, s_3$$
$$P'_{\text{Bob}} = s_2, s_1, s_3$$

DA + improvement cycles is not strategyproof.

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_2	s_1
s_3	s_3	s_2
s_1	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Carol	Carol
Bob	[Alice, Bob]	Bob
Carol		Alice

There are two stable assignments:

$$\mu(\text{Alice}) = s_2, \quad \mu(\text{Bob}) = s_3, \quad \text{and} \quad \mu(\text{Carol}) = s_1 \; .$$
$$\mu'(\text{Alice}) = s_3, \quad \mu'(\text{Bob}) = s_2, \quad \text{and} \quad \mu'(\text{Carol}) = s_1 \; .$$

Let

$$P'_{\text{Alice}} = s_2, s_1, s_3$$
$$P'_{\text{Bob}} = s_2, s_1, s_3$$

DA + improvement cycles is not strategyproof.

P_{Alice}	P_{Bob}	P_{Carol}
s_2	s_2	s_1
s_3	s_3	s_2
s_1	s_1	s_1

$\bar{\pi}_{s_1}$	$\bar{\pi}_{s_2}$	$\bar{\pi}_{s_3}$
Alice	Carol	Carol
Bob	[Alice, Bob]	Bob
Carol		Alice

There are two stable assignments:

$$\mu(\text{Alice}) = s_2, \quad \mu(\text{Bob}) = s_3, \quad \text{and} \quad \mu(\text{Carol}) = s_1 \ .$$
$$\mu'(\text{Alice}) = s_3, \quad \mu'(\text{Bob}) = s_2, \quad \text{and} \quad \mu'(\text{Carol}) = s_1 \ .$$

Let

$$P'_{\text{Alice}} = s_2, s_1, s_3$$
$$P'_{\text{Bob}} = s_2, s_1, s_3$$

For the profile

$$(P'_{\text{Alice}}, P_{\text{Bob}}, P_{\text{Carol}})$$

only μ is stable (not μ').

For the profile

$$(P_{\text{Alice}}, P'_{\text{Bob}}, P_{\text{Carol}})$$

only μ' is stable (not μ).

If the mechanism selects with the true profile:

- $\mu' \Rightarrow$ Alice is better off lying (submitting P'_{Alice}).
- $\mu \Rightarrow$ Bob is better off lying (submitting P'_{Bob}).

For the profile

$$(P'_{\text{Alice}}, P_{\text{Bob}}, P_{\text{Carol}})$$

only μ is stable (not μ').

For the profile

$$(P_{\text{Alice}}, P'_{\text{Bob}}, P_{\text{Carol}})$$

only μ' is stable (not μ).

If the mechanism selects with the true profile:

- $\mu' \Rightarrow$ Alice is better off lying (submitting P'_{Alice}).
- $\mu \Rightarrow$ Bob is better off lying (submitting P'_{Bob}).

For the profile

$$(P'_{\text{Alice}}, P_{\text{Bob}}, P_{\text{Carol}})$$

only μ is stable (not μ').

For the profile

$$(P_{\text{Alice}}, P'_{\text{Bob}}, P_{\text{Carol}})$$

only μ' is stable (not μ).

If the mechanism selects with the true profile:

- $\mu' \Rightarrow$ Alice is better off lying (submitting P'_{Alice}).
- $\mu \Rightarrow$ Bob is better off lying (submitting P'_{Bob}).

For the profile

$$(P'_{\text{Alice}}, P_{\text{Bob}}, P_{\text{Carol}})$$

only μ is stable (not μ').

For the profile

$$(P_{\text{Alice}}, P'_{\text{Bob}}, P_{\text{Carol}})$$

only μ' is stable (not μ).

If the mechanism selects with the true profile:

- $\mu' \Rightarrow$ Alice is better off lying (submitting P'_{Alice}).
- $\mu \Rightarrow$ Bob is better off lying (submitting P'_{Bob}).

For the profile

$$(P'_{\text{Alice}}, P_{\text{Bob}}, P_{\text{Carol}})$$

only μ is stable (not μ').

For the profile

$$(P_{\text{Alice}}, P'_{\text{Bob}}, P_{\text{Carol}})$$

only μ' is stable (not μ).

If the mechanism selects with the true profile:

- $\mu' \Rightarrow$ Alice is better off lying (submitting P'_{Alice}).
- $\mu \Rightarrow$ Bob is better off lying (submitting P'_{Bob}).

How to break ties?

There are two options to break ties:

- Multiple tie-breaking:
  Each school has its own tie-breaking.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - For s_1 Alice ends up with a higher priority than Bob.
  - For s_2 Bob ends up with a higher priority than Alice.

- **Single tie-breaking**:
  The tie-breaking is the same for all schools.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - If at s_1 Alice ends up with a higher priority than Bob,
  - then at s_2 Alice ends up with a higher priority than Bob.

How to break ties?

There are two options to break ties:

- Multiple tie-breaking:
  Each school has its own tie-breaking.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - For s_1 Alice ends up with a higher priority than Bob.
  - For s_2 Bob ends up with a higher priority than Alice.

- **Single tie-breaking**:
  The tie-breaking is the same for all schools.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - If at s_1 Alice ends up with a higher priority than Bob,
  - then at s_2 Alice ends up with a higher priority than Bob.

How to break ties?

There are two options to break ties:

- Multiple tie-breaking:
  Each school has its own tie-breaking.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.

  - For s_1 Alice ends up with a higher priority than Bob.
  - For s_2 Bob ends up with a higher priority than Alice.

- **Single tie-breaking**:
  The tie-breaking is the same for all schools.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.

  - If at s_1 Alice ends up with a higher priority than Bob,
  - then at s_2 Alice ends up with a higher priority than Bob.

How to break ties?

There are two options to break ties:

- Multiple tie-breaking:
  Each school has its own tie-breaking.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - For s_1 Alice ends up with a higher priority than Bob.
  - For s_2 Bob ends up with a higher priority than Alice.

- **Single tie-breaking**:
  The tie-breaking is the same for all schools.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - If at s_1 Alice ends up with a higher priority than Bob,
  - then at s_2 Alice ends up with a higher priority than Bob.

How to break ties?

There are two options to break ties:

- Multiple tie-breaking:
  Each school has its own tie-breaking.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.

  - For s_1 Alice ends up with a higher priority than Bob.
  - For s_2 Bob ends up with a higher priority than Alice.

- **Single tie-breaking**:
  The tie-breaking is the same for all schools.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.

  - If at s_1 Alice ends up with a higher priority than Bob,
  - then at s_2 Alice ends up with a higher priority than Bob.

How to break ties?

There are two options to break ties:

- Multiple tie-breaking:
  Each school has its own tie-breaking.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - For s_1 Alice ends up with a higher priority than Bob.
  - For s_2 Bob ends up with a higher priority than Alice.

- **Single tie-breaking**:
  The tie-breaking is the same for all schools.

  Example: Alice and Bob are in the same tiers for schools s_1 and s_2.
  - If at s_1 Alice ends up with a higher priority than Bob,
  - then at s_2 Alice ends up with a higher priority than Bob.

Multiple tie-breaking seems more fair.

$$\bar{\pi}_s = [\text{Alice}], [\text{Bob}, \text{Carol}, \text{Denis}, \ \text{Erin}, \ \text{Fred}], \text{Gilda}$$

and s has only 1 seat.

If breaking ties gives

$$\pi_s = \text{Alice}, \text{Bob}, \text{Carol} \dots$$

Then if Bob wants s he only needs that Alice is not assigned to s.

But if breaking ties gives

$$\pi'_s = \text{Alice}, \text{Carol}, \ \text{Denis}, \ \text{Erin}, \ \text{Fred}, \ \text{Bob}, \dots$$

Then if Bob wants s he needs that Alice, Carol, Denis, Erin and Fred are not assigned to s .

Multiple tie-breaking seems more fair.

$$\bar{\pi}_s = [\text{Alice}], [\text{Bob}, \text{Carol}, \text{Denis},\ \text{Erin},\ \text{Fred}], \text{Gilda}$$

and s has only 1 seat.

If breaking ties gives

$$\pi_s = \text{Alice}, \text{Bob}, \text{Carol} \dots$$

Then if Bob wants s he only needs that Alice is not assigned to s.

But if breaking ties gives

$$\pi'_s = \text{Alice}, \text{Carol},\ \text{Denis},\ \text{Erin},\ \text{Fred},\ \text{Bob}, \dots$$

Then if Bob wants s he needs that Alice, Carol, Denis, Erin and Fred are not assigned to s .

Multiple tie-breaking seems more fair.

$$\bar{\pi}_s = [\text{Alice}], [\text{Bob}, \text{Carol}, \text{Denis}, \ \text{Erin}, \ \text{Fred}], \text{Gilda}$$

and s has only 1 seat.

If breaking ties gives

$$\pi_s = \text{Alice}, \text{Bob}, \text{Carol} \dots$$

Then if Bob wants s he only needs that Alice is not assigned to s.

But if breaking ties gives

$$\pi'_s = \text{Alice}, \text{Carol}, \ \text{Denis}, \ \text{Erin}, \ \text{Fred}, \ \text{Bob}, \dots$$

Then if Bob wants s he needs that Alice, Carol, Denis, Erin and Fred are not assigned to s .

Multiple tie-breaking seems more fair.

$$\bar{\pi}_s = [\text{Alice}], [\text{Bob}, \text{Carol}, \text{Denis},\ \text{Erin},\ \text{Fred}], \text{Gilda}$$

and s has only 1 seat.

If breaking ties gives

$$\pi_s = \text{Alice}, \text{Bob}, \text{Carol} \dots$$

Then if Bob wants s he only needs that Alice is not assigned to s.

But if breaking ties gives

$$\pi'_s = \text{Alice}, \text{Carol},\ \text{Denis},\ \text{Erin},\ \text{Fred},\ \text{Bob}, \dots$$

Then if Bob wants s he needs that Alice, Carol, Denis, Erin and Fred are not assigned to s .

So, multiple tie-breaking seems to give equal chances to each student.

Theorem

If μ is a stable assignment such that:

- *μ can be obtained using a multiple tie-breaking rule*
- *μ cannot be obtained using a single tie-breaking rule*

then that assignment is not a student-optimal assignment.

$\Rightarrow$ we better use single tie-breaking rules.

So, multiple tie-breaking seems to give equal chances to each student.

Theorem

If μ is a stable assignment such that:

- *μ can be obtained using a multiple tie-breaking rule*
- *μ cannot be obtained using a single tie-breaking rule*

then that assignment is not a student-optimal assignment.

$\Rightarrow$ we better use single tie-breaking rules.

Take-away

- School choice is a many-to-one assignment problem.

  Many insights and results are the same as for the medical match model, but as an assignment problem only students' welfare matter.

- Efficiency and stability are two properties we may want. There are not compatible.

  - Stability can be obtained with the Deferred Acceptance (with students proposing). It is strategyproof.

  - Efficiency can be obtained with the Top Trading Cycle algorithm. It is strategyproof.

- The immediate acceptance algorithm is another possible solution, often used in practice. It produces efficient (but not stable) matchings. It is not strategyproof.

  In general, IA and TTC do not produce the same assignments.

- The city of Boston used IA until 2006. In 2007 it switched to DA to assign students.

  School choice in Boston is an assignment problem: schools do not have preferences over students.

- New York City switched from a decentralized to a centralized matching mechanism, using DA with students proposing.

  School choice in NYC is a matching mechanism: some schools are strategic and have preferences over students.