New algorithms for school choice

Mark C. Wilson
UMass Amherst

QUB Easter workshop
2022-04-25

Some general principles of my research

- Follow your interests and have fun: apparently recreational examples can lead to serious research.
- Axioms are important, and under-used by many. However economics and social choice may be in danger of overusing them.
- Worst-case analysis is important, but overused. It gives nice, clean, but usually rather weak and unrealistic results.
- Average-case analysis is underused. It has its own pitfalls.

Some general principles of my research

- Follow your interests and have fun: apparently recreational examples can lead to serious research.
- Axioms are important, and under-used by many. However economics and social choice may be in danger of overusing them.
- Worst-case analysis is important, but overused. It gives nice, clean, but usually rather weak and unrealistic results.
- Average-case analysis is underused. It has its own pitfalls.

Some general principles of my research

- Follow your interests and have fun: apparently recreational examples can lead to serious research.
- Axioms are important, and under-used by many. However economics and social choice may be in danger of overusing them.
- Worst-case analysis is important, but overused. It gives nice, clean, but usually rather weak and unrealistic results.
- Average-case analysis is underused. It has its own pitfalls.

Some general principles of my research

- Follow your interests and have fun: apparently recreational examples can lead to serious research.
- Axioms are important, and under-used by many. However economics and social choice may be in danger of overusing them.
- Worst-case analysis is important, but overused. It gives nice, clean, but usually rather weak and unrealistic results.
- Average-case analysis is underused. It has its own pitfalls.

Tradeoffs in social choice

- An axiomatic approach to collective decision-making is valuable, but it can be excessively limiting.
- Tradeoffs between multiple criteria (e.g. strategyproofness, efficiency, welfare, simplicity) must be made.
- Axiomatically nice mechanisms often over-optimize a single criterion, corresponding to extreme points in the "space of mechanisms".
- We need to explore the interior of that space.
- Axiomatic approaches use inherently worst case reasoning, when sometimes the average case is more relevant.

Tradeoffs in social choice

- An axiomatic approach to collective decision-making is valuable, but it can be excessively limiting.
- Tradeoffs between multiple criteria (e.g. strategyproofness, efficiency, welfare, simplicity) must be made.
- Axiomatically nice mechanisms often over-optimize a single criterion, corresponding to extreme points in the "space of mechanisms".
- We need to explore the interior of that space.
- Axiomatic approaches use inherently worst case reasoning, when sometimes the average case is more relevant.

Tradeoffs in social choice

- An axiomatic approach to collective decision-making is valuable, but it can be excessively limiting.
- Tradeoffs between multiple criteria (e.g. strategyproofness, efficiency, welfare, simplicity) must be made.
- Axiomatically nice mechanisms often over-optimize a single criterion, corresponding to extreme points in the "space of mechanisms".
- We need to explore the interior of that space.
- Axiomatic approaches use inherently worst case reasoning, when sometimes the average case is more relevant.

Tradeoffs in social choice

- An axiomatic approach to collective decision-making is valuable, but it can be excessively limiting.
- Tradeoffs between multiple criteria (e.g. strategyproofness, efficiency, welfare, simplicity) must be made.
- Axiomatically nice mechanisms often over-optimize a single criterion, corresponding to extreme points in the "space of mechanisms".
- We need to explore the interior of that space.
- Axiomatic approaches use inherently worst case reasoning, when sometimes the average case is more relevant.

Tradeoffs in social choice

- An axiomatic approach to collective decision-making is valuable, but it can be excessively limiting.
- Tradeoffs between multiple criteria (e.g. strategyproofness, efficiency, welfare, simplicity) must be made.
- Axiomatically nice mechanisms often over-optimize a single criterion, corresponding to extreme points in the "space of mechanisms".
- We need to explore the interior of that space.
- Axiomatic approaches use inherently worst case reasoning, when sometimes the average case is more relevant.

Strategyproofness

- Compared to many researchers, I am less concerned about strategyproofness and more about welfare.
  - For voting rules, strategyproofness usually leads to dictatorship, and hence low welfare.
  - Other examples: Serial Dictatorship for house allocation (below) is strategyproof and efficient but sacrifices welfare and particularly fairness.
  - The DA mechanism below for school choice is strategyproof for students but is known to give lower welfare than IA (Boston) under a wide variety of scenarios [4, 6] but not all [2].
- Strategic behavior cannot be ignored, but in many situations (incomplete information, heterogeneous preferences, complex mechanisms) it happens rarely.
- I concentrate on truthful behavior of agents, hence on underlying *allocation algorithms* rather than *mechanisms*.

Strategyproofness

- Compared to many researchers, I am less concerned about strategyproofness and more about welfare.
  - For voting rules, strategyproofness usually leads to dictatorship, and hence low welfare.
  - Other examples: Serial Dictatorship for house allocation (below) is strategyproof and efficient but sacrifices welfare and particularly fairness.
  - The DA mechanism below for school choice is strategyproof for students but is known to give lower welfare than IA (Boston) under a wide variety of scenarios [4, 6] but not all [2].
- Strategic behavior cannot be ignored, but in many situations (incomplete information, heterogeneous preferences, complex mechanisms) it happens rarely.
- I concentrate on truthful behavior of agents, hence on underlying *allocation algorithms* rather than *mechanisms*.

Strategyproofness

- Compared to many researchers, I am less concerned about strategyproofness and more about welfare.
  - For voting rules, strategyproofness usually leads to dictatorship, and hence low welfare.
  - Other examples: Serial Dictatorship for house allocation (below) is strategyproof and efficient but sacrifices welfare and particularly fairness.
  - The DA mechanism below for school choice is strategyproof for students but is known to give lower welfare than IA (Boston) under a wide variety of scenarios [4, 6] but not all [2].
- Strategic behavior cannot be ignored, but in many situations (incomplete information, heterogeneous preferences, complex mechanisms) it happens rarely.
- I concentrate on truthful behavior of agents, hence on underlying *allocation algorithms* rather than *mechanisms*.

Strategyproofness

- Compared to many researchers, I am less concerned about strategyproofness and more about welfare.
  - For voting rules, strategyproofness usually leads to dictatorship, and hence low welfare.
  - Other examples: Serial Dictatorship for house allocation (below) is strategyproof and efficient but sacrifices welfare and particularly fairness.
  - The DA mechanism below for school choice is strategyproof for students but is known to give lower welfare than IA (Boston) under a wide variety of scenarios [4, 6] but not all [2].
- Strategic behavior cannot be ignored, but in many situations (incomplete information, heterogeneous preferences, complex mechanisms) it happens rarely.
- I concentrate on truthful behavior of agents, hence on underlying *allocation algorithms* rather than *mechanisms*.

Strategyproofness

- Compared to many researchers, I am less concerned about strategyproofness and more about welfare.
  - For voting rules, strategyproofness usually leads to dictatorship, and hence low welfare.
  - Other examples: Serial Dictatorship for house allocation (below) is strategyproof and efficient but sacrifices welfare and particularly fairness.
  - The DA mechanism below for school choice is strategyproof for students but is known to give lower welfare than IA (Boston) under a wide variety of scenarios [4, 6] but not all [2].
- Strategic behavior cannot be ignored, but in many situations (incomplete information, heterogeneous preferences, complex mechanisms) it happens rarely.
- I concentrate on truthful behavior of agents, hence on underlying *allocation algorithms* rather than *mechanisms*.

Strategyproofness

- Compared to many researchers, I am less concerned about strategyproofness and more about welfare.
  - For voting rules, strategyproofness usually leads to dictatorship, and hence low welfare.
  - Other examples: Serial Dictatorship for house allocation (below) is strategyproof and efficient but sacrifices welfare and particularly fairness.
  - The DA mechanism below for school choice is strategyproof for students but is known to give lower welfare than IA (Boston) under a wide variety of scenarios [4, 6] but not all [2].
- Strategic behavior cannot be ignored, but in many situations (incomplete information, heterogeneous preferences, complex mechanisms) it happens rarely.
- I concentrate on truthful behavior of agents, hence on underlying *allocation algorithms* rather than *mechanisms*.

A Christmas party game

- A department Christmas party in 2014: Yankee Swap was played.
- Each person brings an item and they are placed in a pile.
- One person takes a gift. The others in turn can either take one from the pile, or steal one from someone else.
- I realized that it was essentially a decentralized allocation algorithm, and the effort to understand it led to some (IMHO) important concepts.
- We need rules to prevent infinite cycling. For example, no one can take a given present again unless the number of presents in the pile has decreased.

A Christmas party game

- A department Christmas party in 2014: Yankee Swap was played.
- Each person brings an item and they are placed in a pile.
- One person takes a gift. The others in turn can either take one from the pile, or steal one from someone else.
- I realized that it was essentially a decentralized allocation algorithm, and the effort to understand it led to some (IMHO) important concepts.
- We need rules to prevent infinite cycling. For example, no one can take a given present again unless the number of presents in the pile has decreased.

A Christmas party game

- A department Christmas party in 2014: Yankee Swap was played.
- Each person brings an item and they are placed in a pile.
- One person takes a gift. The others in turn can either take one from the pile, or steal one from someone else.
- I realized that it was essentially a decentralized allocation algorithm, and the effort to understand it led to some (IMHO) important concepts.
- We need rules to prevent infinite cycling. For example, no one can take a given present again unless the number of presents in the pile has decreased.

A Christmas party game

- A department Christmas party in 2014: Yankee Swap was played.
- Each person brings an item and they are placed in a pile.
- One person takes a gift. The others in turn can either take one from the pile, or steal one from someone else.
- I realized that it was essentially a decentralized allocation algorithm, and the effort to understand it led to some (IMHO) important concepts.
- We need rules to prevent infinite cycling. For example, no one can take a given present again unless the number of presents in the pile has decreased.

A Christmas party game

- A department Christmas party in 2014: Yankee Swap was played.
- Each person brings an item and they are placed in a pile.
- One person takes a gift. The others in turn can either take one from the pile, or steal one from someone else.
- I realized that it was essentially a decentralized allocation algorithm, and the effort to understand it led to some (IMHO) important concepts.
- We need rules to prevent infinite cycling. For example, no one can take a given present again unless the number of presents in the pile has decreased.

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

School choice setup

- Finite sets:
  - $\mathcal{A}$ — students
  - $\mathcal{I}$ — seats
  - $\mathcal{S}$ — schools

  where $\mathcal{S}$ is simply a partition of $\mathcal{I}$.
- We aim to assign each student to a unique seat (and hence a unique school).
- Students have strict preferences over schools, but no preference for seats in a given school.
- Schools have coarse preferences (priority classes) over students. We typically break ties inside these classes via a single tiebreaking order common to all schools. This has welfare consequences [1].

Connection to other allocation problems

- If we use tiebreakers as above, then each school in fact has a strict preference order for students (college admissions).
- The special case where each school has only one seat, and schools have a single priority class, is housing allocation.
- The special case where students have strict preferences over seats is the marriage problem.
- The most-studied mechanism for housing allocation is Serial Dictatorship — agents take turns to choose their best remaining item.
- The most-studied mechanism for the marriage problem is Gale-Shapley (see DA below).

Connection to other allocation problems

- If we use tiebreakers as above, then each school in fact has a strict preference order for students (college admissions).
- The special case where each school has only one seat, and schools have a single priority class, is housing allocation.
- The special case where students have strict preferences over seats is the marriage problem.
- The most-studied mechanism for housing allocation is Serial Dictatorship — agents take turns to choose their best remaining item.
- The most-studied mechanism for the marriage problem is Gale-Shapley (see DA below).

Connection to other allocation problems

- If we use tiebreakers as above, then each school in fact has a strict preference order for students (college admissions).
- The special case where each school has only one seat, and schools have a single priority class, is housing allocation.
- The special case where students have strict preferences over seats is the marriage problem.
- The most-studied mechanism for housing allocation is Serial Dictatorship — agents take turns to choose their best remaining item.
- The most-studied mechanism for the marriage problem is Gale-Shapley (see DA below).

Connection to other allocation problems

- If we use tiebreakers as above, then each school in fact has a strict preference order for students (college admissions).
- The special case where each school has only one seat, and schools have a single priority class, is housing allocation.
- The special case where students have strict preferences over seats is the marriage problem.
- The most-studied mechanism for housing allocation is Serial Dictatorship — agents take turns to choose their best remaining item.
- The most-studied mechanism for the marriage problem is Gale-Shapley (see DA below).

Connection to other allocation problems

- If we use tiebreakers as above, then each school in fact has a strict preference order for students (college admissions).
- The special case where each school has only one seat, and schools have a single priority class, is housing allocation.
- The special case where students have strict preferences over seats is the marriage problem.
- The most-studied mechanism for housing allocation is Serial Dictatorship — agents take turns to choose their best remaining item.
- The most-studied mechanism for the marriage problem is Gale-Shapley (see DA below).

Sequential Deferred Acceptance

- At each step an unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" means that a has not previously proposed to s.
- If s prefers a to some seat occupant b (including the empty occupant), then s replaces b with a.
- Once no unmatched students remain, we terminate.
- Note that there may be a choice of which b to eject.
- The order of proposals is not important, provided preferences are all strict.

Sequential Deferred Acceptance

- At each step an unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" means that a has not previously proposed to s.
- If s prefers a to some seat occupant b (including the empty occupant), then s replaces b with a.
- Once no unmatched students remain, we terminate.
- Note that there may be a choice of which b to eject.
- The order of proposals is not important, provided preferences are all strict.

Sequential Deferred Acceptance

- At each step an unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" means that a has not previously proposed to s.
- If s prefers a to some seat occupant b (including the empty occupant), then s replaces b with a.
- Once no unmatched students remain, we terminate.
- Note that there may be a choice of which b to eject.
- The order of proposals is not important, provided preferences are all strict.

Sequential Deferred Acceptance

- At each step an unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" means that a has not previously proposed to s.
- If s prefers a to some seat occupant b (including the empty occupant), then s replaces b with a.
- Once no unmatched students remain, we terminate.
- Note that there may be a choice of which b to eject.
- The order of proposals is not important, provided preferences are all strict.

Sequential Deferred Acceptance

- At each step an unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" means that a has not previously proposed to s.
- If s prefers a to some seat occupant b (including the empty occupant), then s replaces b with a.
- Once no unmatched students remain, we terminate.
- Note that there may be a choice of which b to eject.
- The order of proposals is not important, provided preferences are all strict.

Sequential Deferred Acceptance

- At each step an unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" means that a has not previously proposed to s.
- If s prefers a to some seat occupant b (including the empty occupant), then s replaces b with a.
- Once no unmatched students remain, we terminate.
- Note that there may be a choice of which b to eject.
- The order of proposals is not important, provided preferences are all strict.

Sequential Immediate Acceptance

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise a goes to the back of the queue.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Immediate Acceptance

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise a goes to the back of the queue.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Immediate Acceptance

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise a goes to the back of the queue.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Immediate Acceptance

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise a goes to the back of the queue.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Immediate Acceptance

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise a goes to the back of the queue.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Immediate Acceptance

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- “Admissible” again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise a goes to the back of the queue.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Serial Dictatorship

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" yet again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise, a approaches schools in decreasing order of preference until accepted.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Serial Dictatorship

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" yet again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise, a approaches schools in decreasing order of preference until accepted.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Serial Dictatorship

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" yet again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise, a approaches schools in decreasing order of preference until accepted.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Serial Dictatorship

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" yet again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise, a approaches schools in decreasing order of preference until accepted.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Serial Dictatorship

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" yet again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise, a approaches schools in decreasing order of preference until accepted.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

Sequential Serial Dictatorship

- Order the students according to the common tiebreaking order above.
- At each step the top unmatched student a proposes to its highest ranked admissible school s.
- "Admissible" yet again means that a has not previously proposed to s.
- If there is an empty seat then a gets one. Otherwise, a approaches schools in decreasing order of preference until accepted.
- Once no unmatched students remain, we terminate.
- The order of proposals is very important.

IA vs SD

- They seem very similar: agents make proposals in turn, and assignments are unbreakable.
- The key difference is that they use a different data structure to control the order of proposals.
- In IA, there is a *queue* containing unmatched agents.
- In SD, there is a *stack*.

IA vs SD

- They seem very similar: agents make proposals in turn, and assignments are unbreakable.
- The key difference is that they use a different data structure to control the order of proposals.
- In IA, there is a *queue* containing unmatched agents.
- In SD, there is a *stack*.

IA vs SD

- They seem very similar: agents make proposals in turn, and assignments are unbreakable.
- The key difference is that they use a different data structure to control the order of proposals.
- In IA, there is a *queue* containing unmatched agents.
- In SD, there is a *stack*.

IA vs SD

- They seem very similar: agents make proposals in turn, and assignments are unbreakable.
- The key difference is that they use a different data structure to control the order of proposals.
- In IA, there is a *queue* containing unmatched agents.
- In SD, there is a *stack*.

A common framework

- The IA and SD algorithms are formally equivalent to a variant of DA where we allow schools to have fictitious dynamic preferences and we impose an order on the proposals.
- In each case, instead of a fixed tiebreaker, we declare that inside each priority class, a student already having a seat is preferred to any proposing student ("Accept-First" protocol).
- Each school prefers every student already having a seat at the school to all other students, so proposals are rejected unless there is an empty seat.
- The only difference is the use of queue or stack to control proposal order.

A common framework

- The IA and SD algorithms are formally equivalent to a variant of DA where we allow schools to have fictitious dynamic preferences and we impose an order on the proposals.
- In each case, instead of a fixed tiebreaker, we declare that inside each priority class, a student already having a seat is preferred to any proposing student ("Accept-First" protocol).
- Each school prefers every student already having a seat at the school to all other students, so proposals are rejected unless there is an empty seat.
- The only difference is the use of queue or stack to control proposal order.

A common framework

- The IA and SD algorithms are formally equivalent to a variant of DA where we allow schools to have fictitious dynamic preferences and we impose an order on the proposals.
- In each case, instead of a fixed tiebreaker, we declare that inside each priority class, a student already having a seat is preferred to any proposing student ("Accept-First" protocol).
- Each school prefers every student already having a seat at the school to all other students, so proposals are rejected unless there is an empty seat.
- The only difference is the use of queue or stack to control proposal order.

A common framework

- The IA and SD algorithms are formally equivalent to a variant of DA where we allow schools to have fictitious dynamic preferences and we impose an order on the proposals.
- In each case, instead of a fixed tiebreaker, we declare that inside each priority class, a student already having a seat is preferred to any proposing student ("Accept-First" protocol).
- Each school prefers every student already having a seat at the school to all other students, so proposals are rejected unless there is an empty seat.
- The only difference is the use of queue or stack to control proposal order.

Another dimension: tiebreaking inside priority classes

- Under IA and SD each school rejects proposals if there are no seats left.
- Instead, we could declare that inside each priority class, a proposing student is always accepted ("Accept-Last"), and we eject an already assigned student to make room, if necessary.
- We would still need to specify *which* student is ejected. A common tiebreak order for all schools is one way.
- This gives two more algorithms analogous to IA and SD, one using a queue and one a stack.
- We have incorporated the stealing aspect of Yankee Swap.

Another dimension: tiebreaking inside priority classes

- Under IA and SD each school rejects proposals if there are no seats left.
- Instead, we could declare that inside each priority class, a proposing student is always accepted ("Accept-Last"), and we eject an already assigned student to make room, if necessary.
- We would still need to specify *which* student is ejected. A common tiebreak order for all schools is one way.
- This gives two more algorithms analogous to IA and SD, one using a queue and one a stack.
- We have incorporated the stealing aspect of Yankee Swap.

Another dimension: tiebreaking inside priority classes

- Under IA and SD each school rejects proposals if there are no seats left.
- Instead, we could declare that inside each priority class, a proposing student is always accepted ("Accept-Last"), and we eject an already assigned student to make room, if necessary.
- We would still need to specify *which* student is ejected. A common tiebreak order for all schools is one way.
- This gives two more algorithms analogous to IA and SD, one using a queue and one a stack.
- We have incorporated the stealing aspect of Yankee Swap.

Another dimension: tiebreaking inside priority classes

- Under IA and SD each school rejects proposals if there are no seats left.
- Instead, we could declare that inside each priority class, a proposing student is always accepted ("Accept-Last"), and we eject an already assigned student to make room, if necessary.
- We would still need to specify *which* student is ejected. A common tiebreak order for all schools is one way.
- This gives two more algorithms analogous to IA and SD, one using a queue and one a stack.
- We have incorporated the stealing aspect of Yankee Swap.

Another dimension: tiebreaking inside priority classes

- Under IA and SD each school rejects proposals if there are no seats left.
- Instead, we could declare that inside each priority class, a proposing student is always accepted ("Accept-Last"), and we eject an already assigned student to make room, if necessary.
- We would still need to specify *which* student is ejected. A common tiebreak order for all schools is one way.
- This gives two more algorithms analogous to IA and SD, one using a queue and one a stack.
- We have incorporated the stealing aspect of Yankee Swap.

Another dimension: temporary memory

- So far we have not fully incorporated Yankee Swap into this framework.
- All algorithms so far have permanent memory. That is, each agent can propose to each school at most once.
- We can relax this to allow temporary memory: the memory of each school to be reset each time the total number of assigned students increases. This allows seat-stealing.
- A school s is now admissible for a if a has not applied to s *since the last reset*.
- This gives 4 new algorithms, so 8 in total.
- We label them by memory (P/T), tiebreak policy (F/L) and data structure (Q/S). Here PFS is SD, PFQ is the Boston mechanism, TLS is Yankee Swap. Try TFS as a party game and let the youngest go last!

Another dimension: temporary memory

- So far we have not fully incorporated Yankee Swap into this framework.
- All algorithms so far have permanent memory. That is, each agent can propose to each school at most once.
- We can relax this to allow temporary memory: the memory of each school to be reset each time the total number of assigned students increases. This allows seat-stealing.
- A school s is now admissible for a if a has not applied to s *since the last reset*.
- This gives 4 new algorithms, so 8 in total.
- We label them by memory (P/T), tiebreak policy (F/L) and data structure (Q/S). Here PFS is SD, PFQ is the Boston mechanism, TLS is Yankee Swap. Try TFS as a party game and let the youngest go last!

Another dimension: temporary memory

- So far we have not fully incorporated Yankee Swap into this framework.
- All algorithms so far have permanent memory. That is, each agent can propose to each school at most once.
- We can relax this to allow temporary memory: the memory of each school to be reset each time the total number of assigned students increases. This allows seat-stealing.
- A school s is now admissible for a if a has not applied to s *since the last reset*.
- This gives 4 new algorithms, so 8 in total.
- We label them by memory (P/T), tiebreak policy (F/L) and data structure (Q/S). Here PFS is SD, PFQ is the Boston mechanism, TLS is Yankee Swap. Try TFS as a party game and let the youngest go last!

Another dimension: temporary memory

- So far we have not fully incorporated Yankee Swap into this framework.
- All algorithms so far have permanent memory. That is, each agent can propose to each school at most once.
- We can relax this to allow temporary memory: the memory of each school to be reset each time the total number of assigned students increases. This allows seat-stealing.
- A school s is now admissible for a if a has not applied to s *since the last reset*.
- This gives 4 new algorithms, so 8 in total.
- We label them by memory (P/T), tiebreak policy (F/L) and data structure (Q/S). Here PFS is SD, PFQ is the Boston mechanism, TLS is Yankee Swap. Try TFS as a party game and let the youngest go last!

Another dimension: temporary memory

- So far we have not fully incorporated Yankee Swap into this framework.
- All algorithms so far have permanent memory. That is, each agent can propose to each school at most once.
- We can relax this to allow temporary memory: the memory of each school to be reset each time the total number of assigned students increases. This allows seat-stealing.
- A school s is now admissible for a if a has not applied to s *since the last reset*.
- This gives 4 new algorithms, so 8 in total.
- We label them by memory (P/T), tiebreak policy (F/L) and data structure (Q/S). Here PFS is SD, PFQ is the Boston mechanism, TLS is Yankee Swap. Try TFS as a party game and let the youngest go last!

Another dimension: temporary memory

- So far we have not fully incorporated Yankee Swap into this framework.
- All algorithms so far have permanent memory. That is, each agent can propose to each school at most once.
- We can relax this to allow temporary memory: the memory of each school to be reset each time the total number of assigned students increases. This allows seat-stealing.
- A school s is now admissible for a if a has not applied to s *since the last reset*.
- This gives 4 new algorithms, so 8 in total.
- We label them by memory (P/T), tiebreak policy (F/L) and data structure (Q/S). Here PFS is SD, PFQ is the Boston mechanism, TLS is Yankee Swap. Try TFS as a party game and let the youngest go last!

Example (Housing allocation for simplicity)

Agents $1, 2, 3$ have preferences $a \succ b \succ c \succ d$ and agent 4 has $b \succ a \succ c \succ d$.

Algorithm	Allocation	Ranks	Number of proposals
PFS	$1 : a, 2 : b, 3 : c, 4 : d$	(1,2,3,4)	10
PFQ	$1 : a, 2 : c, 3 : d, 4 : b$	(1,3,4,1)	9
PLS	$1 : d, 2 : c, 3 : a, 4 : b$	(4,3,1,1)	9
PLQ	$1 : d, 2 : c, 3 : b, 4 : a$	(4,3,2,2)	10
TFS	$1 : d, 2 : a, 3 : c, 4 : b$	(4,1,3,1)	15
TFQ	$1 : a, 2 : b, 3 : d, 4 : c$	(1,2,4,3)	33
TLS	$1 : b, 2 : a, 3 : d, 4 : c$	(2,1,4,3)	16
TLQ	$1 : a, 2 : b, 3 : d, 4 : c$	(1,2,4,3)	21

Summary of properties

- The “F” algorithms are student-efficient, and the “L” ones are not.
- Running TTC on the output of the “L” algorithms yields somewhat better welfare than the “F” ones.
- The “Q” algorithms seem much fairer to those later in the tiebreak order than the “S” ones. They can also be implemented using rounds with simultaneous proposals.
- The “T” algorithms seem a little better than the “P” ones in terms of utilitarian welfare and justified envy. They take longer to run, but still low degree polynomial time.

Summary of properties

- The "F" algorithms are student-efficient, and the "L" ones are not.
- Running TTC on the output of the "L" algorithms yields somewhat better welfare than the "F" ones.
- The "Q" algorithms seem much fairer to those later in the tiebreak order than the "S" ones. They can also be implemented using rounds with simultaneous proposals.
- The "T" algorithms seem a little better than the "P" ones in terms of utilitarian welfare and justified envy. They take longer to run, but still low degree polynomial time.

Summary of properties

- The “F” algorithms are student-efficient, and the “L” ones are not.
- Running TTC on the output of the “L” algorithms yields somewhat better welfare than the “F” ones.
- The “Q” algorithms seem much fairer to those later in the tiebreak order than the “S” ones. They can also be implemented using rounds with simultaneous proposals.
- The “T” algorithms seem a little better than the “P” ones in terms of utilitarian welfare and justified envy. They take longer to run, but still low degree polynomial time.

Summary of properties

- The “F” algorithms are student-efficient, and the “L” ones are not.
- Running TTC on the output of the “L” algorithms yields somewhat better welfare than the “F” ones.
- The “Q” algorithms seem much fairer to those later in the tiebreak order than the “S” ones. They can also be implemented using rounds with simultaneous proposals.
- The “T” algorithms seem a little better than the “P” ones in terms of utilitarian welfare and justified envy. They take longer to run, but still low degree polynomial time.

Order symmetry

- We define order symmetry, an average-case fairness concept in this ordinal setting: each agent has equal chance of getting their first choice, equal chance of their second item, etc.
- Let P be a probability measure on preferences. An assignment algorithm is *order symmetric* with respect to P if the expected rank distribution matrix with respect to P has all rows equal.
- This is a concept of fairness at a point in time after the roles of the agents in the algorithm have been determined but before preferences are known.
- Define *order bias* (with respect to a given scoring rule) to be the difference between maximum and minimum expected utility.
- For example, for SD in the housing allocation setup, the first agent always gets their top choice and the last has to take whatever is left, so order bias is (almost) the average score.

Order symmetry

- We define order symmetry, an average-case fairness concept in this ordinal setting: each agent has equal chance of getting their first choice, equal chance of their second item, etc.
- Let P be a probability measure on preferences. An assignment algorithm is *order symmetric* with respect to P if the expected rank distribution matrix with respect to P has all rows equal.
- This is a concept of fairness at a point in time after the roles of the agents in the algorithm have been determined but before preferences are known.
- Define *order bias* (with respect to a given scoring rule) to be the difference between maximum and minimum expected utility.
- For example, for SD in the housing allocation setup, the first agent always gets their top choice and the last has to take whatever is left, so order bias is (almost) the average score.

Order symmetry

- We define order symmetry, an average-case fairness concept in this ordinal setting: each agent has equal chance of getting their first choice, equal chance of their second item, etc.
- Let P be a probability measure on preferences. An assignment algorithm is *order symmetric* with respect to P if the expected rank distribution matrix with respect to P has all rows equal.
- This is a concept of fairness at a point in time after the roles of the agents in the algorithm have been determined but before preferences are known.
- Define *order bias* (with respect to a given scoring rule) to be the difference between maximum and minimum expected utility.
- For example, for SD in the housing allocation setup, the first agent always gets their top choice and the last has to take whatever is left, so order bias is (almost) the average score.

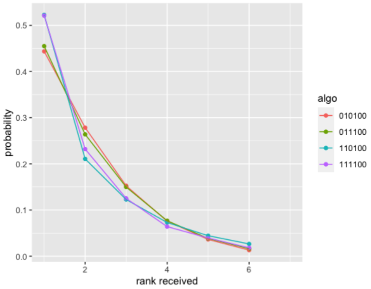
Order symmetry

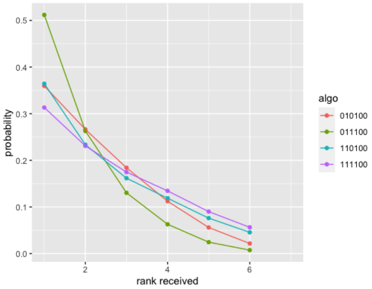
- We define order symmetry, an average-case fairness concept in this ordinal setting: each agent has equal chance of getting their first choice, equal chance of their second item, etc.
- Let P be a probability measure on preferences. An assignment algorithm is *order symmetric* with respect to P if the expected rank distribution matrix with respect to P has all rows equal.
- This is a concept of fairness at a point in time after the roles of the agents in the algorithm have been determined but before preferences are known.
- Define *order bias* (with respect to a given scoring rule) to be the difference between maximum and minimum expected utility.
- For example, for SD in the housing allocation setup, the first agent always gets their top choice and the last has to take whatever is left, so order bias is (almost) the average score.

Order symmetry

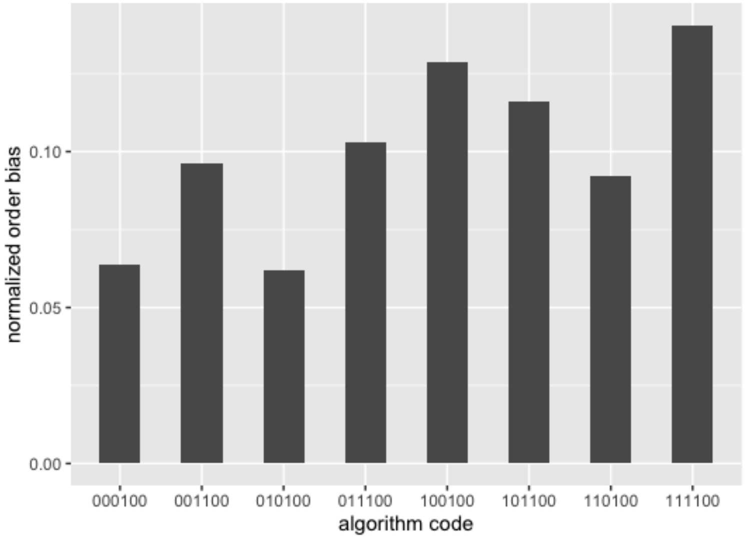
- We define order symmetry, an average-case fairness concept in this ordinal setting: each agent has equal chance of getting their first choice, equal chance of their second item, etc.
- Let P be a probability measure on preferences. An assignment algorithm is *order symmetric* with respect to P if the expected rank distribution matrix with respect to P has all rows equal.
- This is a concept of fairness at a point in time after the roles of the agents in the algorithm have been determined but before preferences are known.
- Define *order bias* (with respect to a given scoring rule) to be the difference between maximum and minimum expected utility.
- For example, for SD in the housing allocation setup, the first agent always gets their top choice and the last has to take whatever is left, so order bias is (almost) the average score.

Fate of the first and last agent, $n = 64$, 6 schools

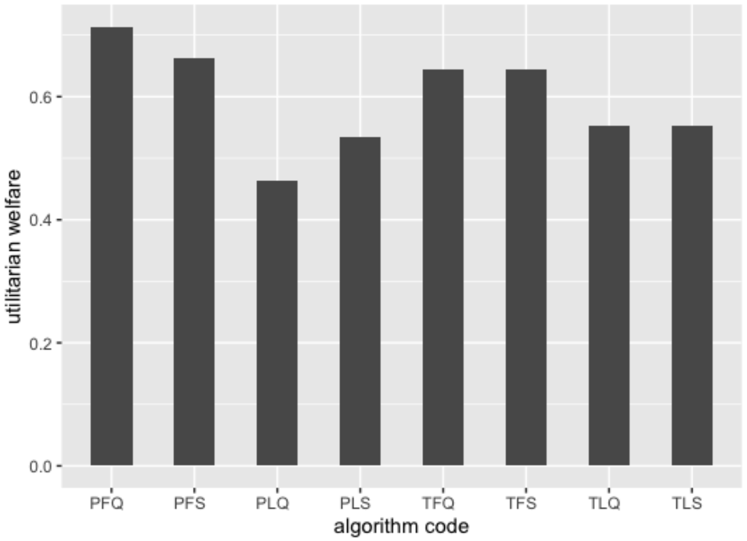




Order bias for same data using common linear utilities (Borda rule)



Utilitarian welfare (house allocation, $n = 20$, Mallows 0.6)



Further developments

- Freeman, Pritchard & Wilson have formalized order bias and order symmetry [3].
- Pritchard & Wilson have derived asymptotic results for Boston in the housing allocation model [5].
- Independently, Zick (2022) has used the Yankee Swap idea (with a *priority queue* to order the proposals) for university course allocation, and it seems very promising.
- There are still many adaptations we could make to the above algorithms.
- We have not yet studied the effect of randomizing the original order, or done a really thorough simulation analysis, or looked at real data.

Further developments

- Freeman, Pritchard & Wilson have formalized order bias and order symmetry [3].
- Pritchard & Wilson have derived asymptotic results for Boston in the housing allocation model [5].
- Independently, Zick (2022) has used the Yankee Swap idea (with a *priority queue* to order the proposals) for university course allocation, and it seems very promising.
- There are still many adaptations we could make to the above algorithms.
- We have not yet studied the effect of randomizing the original order, or done a really thorough simulation analysis, or looked at real data.

Further developments

- Freeman, Pritchard & Wilson have formalized order bias and order symmetry [3].
- Pritchard & Wilson have derived asymptotic results for Boston in the housing allocation model [5].
- Independently, Zick (2022) has used the Yankee Swap idea (with a *priority queue* to order the proposals) for university course allocation, and it seems very promising.
- There are still many adaptations we could make to the above algorithms.
- We have not yet studied the effect of randomizing the original order, or done a really thorough simulation analysis, or looked at real data.

Further developments

- Freeman, Pritchard & Wilson have formalized order bias and order symmetry [3].
- Pritchard & Wilson have derived asymptotic results for Boston in the housing allocation model [5].
- Independently, Zick (2022) has used the Yankee Swap idea (with a *priority queue* to order the proposals) for university course allocation, and it seems very promising.
- There are still many adaptations we could make to the above algorithms.
- We have not yet studied the effect of randomizing the original order, or done a really thorough simulation analysis, or looked at real data.

Further developments

- Freeman, Pritchard & Wilson have formalized order bias and order symmetry [3].
- Pritchard & Wilson have derived asymptotic results for Boston in the housing allocation model [5].
- Independently, Zick (2022) has used the Yankee Swap idea (with a *priority queue* to order the proposals) for university course allocation, and it seems very promising.
- There are still many adaptations we could make to the above algorithms.
- We have not yet studied the effect of randomizing the original order, or done a really thorough simulation analysis, or looked at real data.

Takeaways

- We have a unified presentation of a family of algorithms including known ones that were not connected before.
- The idea of fictitious dynamic preferences systematically connects one-sided and two-sided matching, and also school choice. This may allow more interesting algorithms to be discovered.
- We should investigate far more social choice mechanisms than we have been doing. Having a greater stock of simply described algorithms allows more flexibility.
- Interesting research can come from any application, and we should be on the lookout.
- Simple concrete examples can lead to important theoretical concepts (we think order symmetry is such a concept).

Takeaways

- We have a unified presentation of a family of algorithms including known ones that were not connected before.
- The idea of fictitious dynamic preferences systematically connects one-sided and two-sided matching, and also school choice. This may allow more interesting algorithms to be discovered.
- We should investigate far more social choice mechanisms than we have been doing. Having a greater stock of simply described algorithms allows more flexibility.
- Interesting research can come from any application, and we should be on the lookout.
- Simple concrete examples can lead to important theoretical concepts (we think order symmetry is such a concept).

Takeaways

- We have a unified presentation of a family of algorithms including known ones that were not connected before.
- The idea of fictitious dynamic preferences systematically connects one-sided and two-sided matching, and also school choice. This may allow more interesting algorithms to be discovered.
- We should investigate far more social choice mechanisms than we have been doing. Having a greater stock of simply described algorithms allows more flexibility.
- Interesting research can come from any application, and we should be on the lookout.
- Simple concrete examples can lead to important theoretical concepts (we think order symmetry is such a concept).

Takeaways

- We have a unified presentation of a family of algorithms including known ones that were not connected before.
- The idea of fictitious dynamic preferences systematically connects one-sided and two-sided matching, and also school choice. This may allow more interesting algorithms to be discovered.
- We should investigate far more social choice mechanisms than we have been doing. Having a greater stock of simply described algorithms allows more flexibility.
- Interesting research can come from any application, and we should be on the lookout.
- Simple concrete examples can lead to important theoretical concepts (we think order symmetry is such a concept).

Takeaways

- We have a unified presentation of a family of algorithms including known ones that were not connected before.
- The idea of fictitious dynamic preferences systematically connects one-sided and two-sided matching, and also school choice. This may allow more interesting algorithms to be discovered.
- We should investigate far more social choice mechanisms than we have been doing. Having a greater stock of simply described algorithms allows more flexibility.
- Interesting research can come from any application, and we should be on the lookout.
- Simple concrete examples can lead to important theoretical concepts (we think order symmetry is such a concept).

References I

Aytek Erdil and Haluk Ergin, *What's the matter with tie-breaking? Improving efficiency in school choice*, American Economic Review **98** (2008), no. 3, 669–89.

Haluk Ergin and Tayfun Sönmez, *Games of school choice under the Boston mechanism*, Journal of Public Economics **90** (2006), no. 1-2, 215–237.

Rupert Freeman, Geoffrey Pritchard, and Mark C. Wilson, *Order symmetry: A new fairness criterion for assignment mechanisms*, Jul 2021.

Onur Kesten, *School choice with consent*, The Quarterly Journal of Economics **125** (2010), no. 3, 1297–1348.

Geoffrey Pritchard and Mark C. Wilson, *Asymptotic welfare performance of Boston assignment algorithms*, (2022).

References II

Peter Troyan, *Comparing school choice mechanisms by interim and ex-ante welfare*, Games and Economic Behavior **75** (2012), no. 2, 936–947.