Average-case analysis of random assignment algorithms

Mark C. Wilson
www.cs.auckland.ac.nz/~mcw/blog/

Department of Computer Science
University of Auckland

CMSS Seminar, 2016-05-02

Background

- This talk is based on joint work with Jacky Lo (summer scholarship student 2015-16).

Background

- This talk is based on joint work with Jacky Lo (summer scholarship student 2015-16).
- Preprint: `https://www.cs.auckland.ac.nz/~mcw/Research/Outputs/LoWi2016.pdf`.

The house allocation problem

- Given a set $A = \{a_1, \ldots, a_n\}$ of items and a set $V = \{v_1, \ldots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.

The house allocation problem

- Given a set $A = \{a_1, \dots, a_n\}$ of items and a set $V = \{v_1, \dots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).

The house allocation problem

- Given a set $A = \{a_1, \ldots, a_n\}$ of items and a set $V = \{v_1, \ldots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).
- The allocation can be done using a discrete assignment (no item may be divided) or a random assignment.

The house allocation problem

- Given a set $A = \{a_1, \ldots, a_n\}$ of items and a set $V = \{v_1, \ldots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).
- The allocation can be done using a discrete assignment (no item may be divided) or a random assignment.
- A random assignment can be interpreted in several ways:

The house allocation problem

- Given a set $A = \{a_1, \ldots, a_n\}$ of items and a set $V = \{v_1, \ldots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).
- The allocation can be done using a discrete assignment (no item may be divided) or a random assignment.
- A random assignment can be interpreted in several ways:
  - divide the objects;

The house allocation problem

- Given a set $A = \{a_1, \ldots, a_n\}$ of items and a set $V = \{v_1, \ldots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).
- The allocation can be done using a discrete assignment (no item may be divided) or a random assignment.
- A random assignment can be interpreted in several ways:
  - divide the objects;
  - time-sharing of objects;

The house allocation problem

- Given a set $A = \{a_1, \dots, a_n\}$ of items and a set $V = \{v_1, \dots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).
- The allocation can be done using a discrete assignment (no item may be divided) or a random assignment.
- A random assignment can be interpreted in several ways:
  - divide the objects;
  - time-sharing of objects;
  - a lottery over the objects.

The house allocation problem

- Given a set $A = \{a_1, \ldots, a_n\}$ of items and a set $V = \{v_1, \ldots, v_n\}$ of agents having preferences over the items, we want to allocate items to agents.
- This is the house allocation problem formalised by Hylland & Zeckhauser (1979).
- The allocation can be done using a discrete assignment (no item may be divided) or a random assignment.
- A random assignment can be interpreted in several ways:
  - divide the objects;
  - time-sharing of objects;
  - a lottery over the objects.
- As usual one can consider this as a decentralized mechanism (allowing for strategic behaviour) or simply study the behaviour when preferences are given sincerely.

Axioms — discrete assignments

- A discrete assignment is (Pareto) efficient if it is not possible to improve any agent's outcome without worsening another agent's outcome.

Axioms — discrete assignments

- A discrete assignment is (Pareto) efficient if it is not possible to improve any agent's outcome without worsening another agent's outcome.
- A discrete assignment is envy-free if there do not exist agents i and j so that i prefers j's allocation to her own.

Axioms — discrete assignments

- A discrete assignment is (Pareto) efficient if it is not possible to improve any agent's outcome without worsening another agent's outcome.
- A discrete assignment is envy-free if there do not exist agents i and j so that i prefers j's allocation to her own.
- If agents have identical preferences, then no discrete assignment can be envy-free.

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:
  - ex post: after realisation of the random choices;

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:
  - ex post: after realisation of the random choices;
  - SD: stochastic dominance of probability distributions;

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:
  - ex post: after realisation of the random choices;
  - SD: stochastic dominance of probability distributions;
  - ex ante: expected utility dominance of probability distributions.

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:
  - ex post: after realisation of the random choices;
  - SD: stochastic dominance of probability distributions;
  - ex ante: expected utility dominance of probability distributions.
- Ex ante efficiency is stronger than SD, which is stronger than ex post.

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:
  - ex post: after realisation of the random choices;
  - SD: stochastic dominance of probability distributions;
  - ex ante: expected utility dominance of probability distributions.
- Ex ante efficiency is stronger than SD, which is stronger than ex post.
- The reverse is true for envy-freeness.

Axioms — random assignments

- When comparing random assignments, there are 3 main ideas:
  - ex post: after realisation of the random choices;
  - SD: stochastic dominance of probability distributions;
  - ex ante: expected utility dominance of probability distributions.
- Ex ante efficiency is stronger than SD, which is stronger than ex post.
- The reverse is true for envy-freeness.
- There are new fairness concepts (weak envy-freeness, proportionality) that have no analogue in the discrete case. We do not discuss them here.

Example — comparing random assignments

- Suppose we have two agents of type $abcd$ and two of type $badc$.

Example — comparing random assignments

- Suppose we have two agents of type $abcd$ and two of type $badc$.
- Consider the random assignments (agents are rows, items are columns)

$$A = \begin{bmatrix} \frac{5}{12} & \frac{1}{12} & \frac{5}{12} & \frac{1}{12} \\ \frac{5}{12} & \frac{1}{12} & \frac{5}{12} & \frac{1}{12} \\ \frac{1}{12} & \frac{5}{12} & \frac{1}{12} & \frac{5}{12} \\ \frac{1}{12} & \frac{5}{12} & \frac{1}{12} & \frac{5}{12} \end{bmatrix} \qquad B = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}.$$

Example — comparing random assignments

- Suppose we have two agents of type $abcd$ and two of type $badc$.
- Consider the random assignments (agents are rows, items are columns)

$$A = \begin{bmatrix} \frac{5}{12} & \frac{1}{12} & \frac{5}{12} & \frac{1}{12} \\ \frac{5}{12} & \frac{1}{12} & \frac{5}{12} & \frac{1}{12} \\ \frac{1}{12} & \frac{5}{12} & \frac{1}{12} & \frac{5}{12} \\ \frac{1}{12} & \frac{5}{12} & \frac{1}{12} & \frac{5}{12} \end{bmatrix} \qquad B = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}.$$

- Then each row of B stochastically dominates its counterpart in A. Thus A is not SD-efficient.

The housing market problem

- This is similar to house allocation, but we assume each agent starts with a house. We seek a decentralised mechanism which gives an efficient outcome, and is strategyproof (no agent has incentive to lie about preferences if all other agents are truthful; everyone being truthful is a Nash equilibrium).

The housing market problem

- This is similar to house allocation, but we assume each agent starts with a house. We seek a decentralised mechanism which gives an efficient outcome, and is strategyproof (no agent has incentive to lie about preferences if all other agents are truthful; everyone being truthful is a Nash equilibrium).
- The answer is given by Gale's Top Trading Cycle mechanism (Scarf & Shapley 1974).

The housing market problem

- This is similar to house allocation, but we assume each agent starts with a house. We seek a decentralised mechanism which gives an efficient outcome, and is strategyproof (no agent has incentive to lie about preferences if all other agents are truthful; everyone being truthful is a Nash equilibrium).
- The answer is given by Gale's Top Trading Cycle mechanism (Scarf & Shapley 1974).
- Each agent i points to the agent currently owning i's top choice. The resulting directed graph has cycles, and we reallocate along the cycle. If cycles remain, repeat with second choices, etc.

The housing market problem

- This is similar to house allocation, but we assume each agent starts with a house. We seek a decentralised mechanism which gives an efficient outcome, and is strategyproof (no agent has incentive to lie about preferences if all other agents are truthful; everyone being truthful is a Nash equilibrium).
- The answer is given by Gale's Top Trading Cycle mechanism (Scarf & Shapley 1974).
- Each agent i points to the agent currently owning i's top choice. The resulting directed graph has cycles, and we reallocate along the cycle. If cycles remain, repeat with second choices, etc.
- If we start with an initial allocation of items and then apply TTC, this gives an algorithm for housing allocation.

The housing market problem

- This is similar to house allocation, but we assume each agent starts with a house. We seek a decentralised mechanism which gives an efficient outcome, and is strategyproof (no agent has incentive to lie about preferences if all other agents are truthful; everyone being truthful is a Nash equilibrium).
- The answer is given by Gale's Top Trading Cycle mechanism (Scarf & Shapley 1974).
- Each agent i points to the agent currently owning i's top choice. The resulting directed graph has cycles, and we reallocate along the cycle. If cycles remain, repeat with second choices, etc.
- If we start with an initial allocation of items and then apply TTC, this gives an algorithm for housing allocation.
- Another general method for constructing algorithms: form a convex combination of the outputs of known algorithms.

Famous algorithms - RP

- The Serial Dictatorship (SD) algorithm (with respect to a given fixed order of agents) simply lets them choose one item at a time until items are exhausted. It is always efficient.

Famous algorithms - RP

- The Serial Dictatorship (SD) algorithm (with respect to a given fixed order of agents) simply lets them choose one item at a time until items are exhausted. It is always efficient.
- The Random Serial Dictatorship (or Random Priority) chooses a random permutation of the agents, then applies SD. It is equivalent to choosing a random assignment, then doing TTC.

Famous algorithms - RP

- The Serial Dictatorship (SD) algorithm (with respect to a given fixed order of agents) simply lets them choose one item at a time until items are exhausted. It is always efficient.
- The Random Serial Dictatorship (or Random Priority) chooses a random permutation of the agents, then applies SD. It is equivalent to choosing a random assignment, then doing TTC.
- RP satisfies some nice axiomatic properties: ex-post efficiency, ex-ante strategyproofness, symmetry.

Famous algorithms - RP

- The Serial Dictatorship (SD) algorithm (with respect to a given fixed order of agents) simply lets them choose one item at a time until items are exhausted. It is always efficient.
- The Random Serial Dictatorship (or Random Priority) chooses a random permutation of the agents, then applies SD. It is equivalent to choosing a random assignment, then doing TTC.
- RP satisfies some nice axiomatic properties: ex-post efficiency, ex-ante strategyproofness, symmetry.
- SD runs in polynomial time; RP is super-exponential owing to the $n!$ permutations, and there is unlikely to be a way around this, because determining whether agent i gets item j with positive probability is an NP-hard problem.

Famous algorithms - PS

- The Probabilistic Serial algorithm constructs a random assignment as follows. Each agent starts to consume its most preferred item, all agents “eating” at unit speed. Whenever an item is completely consumed, each agent currently consuming it moves to its next most preferred item. Stop when all items are consumed.

Famous algorithms - PS

- The Probabilistic Serial algorithm constructs a random assignment as follows. Each agent starts to consume its most preferred item, all agents "eating" at unit speed. Whenever an item is completely consumed, each agent currently consuming it moves to its next most preferred item. Stop when all items are consumed.
- PS satisfies some nice axiomatic properties:

Famous algorithms - PS

- The Probabilistic Serial algorithm constructs a random assignment as follows. Each agent starts to consume its most preferred item, all agents “eating” at unit speed. Whenever an item is completely consumed, each agent currently consuming it moves to its next most preferred item. Stop when all items are consumed.
- PS satisfies some nice axiomatic properties:
  - SD-efficiency

Famous algorithms - PS

- The Probabilistic Serial algorithm constructs a random assignment as follows. Each agent starts to consume its most preferred item, all agents "eating" at unit speed. Whenever an item is completely consumed, each agent currently consuming it moves to its next most preferred item. Stop when all items are consumed.
- PS satisfies some nice axiomatic properties:
  - SD-efficiency
  - SD-strategyproofness

Famous algorithms - PS

- The Probabilistic Serial algorithm constructs a random assignment as follows. Each agent starts to consume its most preferred item, all agents "eating" at unit speed. Whenever an item is completely consumed, each agent currently consuming it moves to its next most preferred item. Stop when all items are consumed.
- PS satisfies some nice axiomatic properties:
  - SD-efficiency
  - SD-strategyproofness
  - symmetry

Famous algorithms - PS

- The Probabilistic Serial algorithm constructs a random assignment as follows. Each agent starts to consume its most preferred item, all agents "eating" at unit speed. Whenever an item is completely consumed, each agent currently consuming it moves to its next most preferred item. Stop when all items are consumed.
- PS satisfies some nice axiomatic properties:
  - SD-efficiency
  - SD-strategyproofness
  - symmetry
- PS runs in polynomial time.

New algorithms — Yankee Swap

- This is based on the party game also known as “White Elephant”.

New algorithms — Yankee Swap

- This is based on the party game also known as "White Elephant".
- Fix an order on the agents (randomize as for RP to get a fairer method). At each round, the next agent i without an item chooses her most preferred one. If that had previously been allocated to another agent j, then i steals it, and j can proceed to choose her most preferred item, etc. We prevent cycling during a round by requiring, for example, that no item can be held by any agent more than once per round.

New algorithms — Yankee Swap

- This is based on the party game also known as "White Elephant".
- Fix an order on the agents (randomize as for RP to get a fairer method). At each round, the next agent i without an item chooses her most preferred one. If that had previously been allocated to another agent j, then i steals it, and j can proceed to choose her most preferred item, etc. We prevent cycling during a round by requiring, for example, that no item can be held by any agent more than once per round.
- YS does not satisfy efficiency, strategyproofness or any kind of envy-freeness.

New algorithms — Yankee Swap

- This is based on the party game also known as "White Elephant".
- Fix an order on the agents (randomize as for RP to get a fairer method). At each round, the next agent i without an item chooses her most preferred one. If that had previously been allocated to another agent j, then i steals it, and j can proceed to choose her most preferred item, etc. We prevent cycling during a round by requiring, for example, that no item can be held by any agent more than once per round.
- YS does not satisfy efficiency, strategyproofness or any kind of envy-freeness.
- YS runs in polynomial time for a fixed order of agents.

New algorithms — Boston mechanism

- This is based on the school assignment algorithm, in which items have priorities (weak preferences) over agents.

New algorithms — Boston mechanism

- This is based on the school assignment algorithm, in which items have priorities (weak preferences) over agents.
- Fix an order on the agents (randomize as for RP to get a fairer method). Assignments are final and agents are removed from consideration once they receive an item. At the first round, each item is assigned to the first remaining agent who ranks it first, if such agent exists. At the next round, second preferences are considered, etc.

New algorithms — Boston mechanism

- This is based on the school assignment algorithm, in which items have priorities (weak preferences) over agents.
- Fix an order on the agents (randomize as for RP to get a fairer method). Assignments are final and agents are removed from consideration once they receive an item. At the first round, each item is assigned to the first remaining agent who ranks it first, if such agent exists. At the next round, second preferences are considered, etc.
- NB does not satisfy efficiency, strategyproofness or any kind of envy-freeness.

New algorithms — Boston mechanism

- This is based on the school assignment algorithm, in which items have priorities (weak preferences) over agents.
- Fix an order on the agents (randomize as for RP to get a fairer method). Assignments are final and agents are removed from consideration once they receive an item. At the first round, each item is assigned to the first remaining agent who ranks it first, if such agent exists. At the next round, second preferences are considered, etc.
- NB does not satisfy efficiency, strategyproofness or any kind of envy-freeness.
- NB runs in polynomial time for a fixed order of agents.

Example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac.
  Profile E': same agents, preferences abc, abc, abc.

Example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac. Profile E': same agents, preferences abc, abc, abc.
- YS proceeds as follows on E given order $1, 2, 3$ on agents: $1 : a; 2 : a, 1 : b; 3 : b, 1 : a, 2 : b, 3 : a, 1 : c$. However on E' it yields $1 : a; 2 : a, 1 : b; 3 : a, 2 : b, 1 : a, 3 : b, 2 : c$.

Example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac. Profile E': same agents, preferences abc, abc, abc.
- YS proceeds as follows on E given order $1, 2, 3$ on agents: $1 : a; 2 : a, 1 : b; 3 : b, 1 : a, 2 : b, 3 : a, 1 : c$. However on E' it yields $1 : a; 2 : a, 1 : b; 3 : a, 2 : b, 1 : a, 3 : b, 2 : c$.
- If E' describes sincere preferences then agent 3 is incentivized to pretend bac, so YS is not strategyproof.

Example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac. Profile E': same agents, preferences abc, abc, abc.
- YS proceeds as follows on E given order $1, 2, 3$ on agents: $1 : a; 2 : a, 1 : b; 3 : b, 1 : a, 2 : b, 3 : a, 1 : c$. However on E' it yields $1 : a; 2 : a, 1 : b; 3 : a, 2 : b, 1 : a, 3 : b, 2 : c$.
- If E' describes sincere preferences then agent 3 is incentivized to pretend bac, so YS is not strategyproof.
- NB proceeds as follows on E given order $1, 2, 3$ on agents: $a : 1, 3 : b; 2 : c$. However on E' it yields $a : 1; b : 2; c : 3$.

Example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac. Profile E': same agents, preferences abc, abc, abc.
- YS proceeds as follows on E given order $1, 2, 3$ on agents: $1 : a; 2 : a, 1 : b; 3 : b, 1 : a, 2 : b, 3 : a, 1 : c$. However on E' it yields $1 : a; 2 : a, 1 : b; 3 : a, 2 : b, 1 : a, 3 : b, 2 : c$.
- If E' describes sincere preferences then agent 3 is incentivized to pretend bac, so YS is not strategyproof.
- NB proceeds as follows on E given order $1, 2, 3$ on agents: $a : 1, 3 : b; 2 : c$. However on E' it yields $a : 1; b : 2; c : 3$.
- If E' describes sincere preferences then agent 3 is incentivized to pretend bac, so NB is not strategyproof.

Example - TTC

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac.

Example - TTC

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac.
- Initial allocation $1 : c, 2 : b, 3 : a$.

Example - TTC

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac.
- Initial allocation $1 : c, 2 : b, 3 : a$.
- Agents 1 and 2 point to 3 and 3 points to 2. A trade between 2 and 3 is mutually beneficial and removes all cycles.

Example - TTC

- Profile E: 3 agents $1, 2, 3$, with preferences abc, abc, bac.
- Initial allocation $1 : c, 2 : b, 3 : a$.
- Agents 1 and 2 point to 3 and 3 points to 2. A trade between 2 and 3 is mutually beneficial and removes all cycles.
- Thus TTC output the allocation $1 : c, 2 : a, 3 : b$.

Example continued

Averaging over all 6 orders of agents, we obtain the random assignment matrices. Note that YS is stochastically dominated by NB for every agent.

$$\begin{bmatrix} 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \quad \text{(YS)}$$

$$\begin{bmatrix} 1/2 & 0 & 1/2 \\ 1/2 & 0 & 1/2 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{(NB and YS+TTC)}$$

$$\begin{bmatrix} 1/2 & 1/6 & 1/3 \\ 1/2 & 1/6 & 1/3 \\ 0 & 2/3 & 1/3 \end{bmatrix} \quad \text{(RP and PS)}$$

Another example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, cba, acb.

Another example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, cba, acb.
- SD yields $1 : a, 2 : c, 3 : b$, as does NB.

Another example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, cba, acb.
- SD yields $1 : a, 2 : c, 3 : b$, as does NB.
- YS yields $1 : b, 2 : c, 3 : a$.

Another example

- Profile E: 3 agents $1, 2, 3$, with preferences abc, cba, acb.
- SD yields $1 : a, 2 : c, 3 : b$, as does NB.
- YS yields $1 : b, 2 : c, 3 : a$.
- Note that it is possible for everyone to obtain one of their top two choices $(1 : b, 2 : c, 3 : a)$, but only YS actually does that.

Our idea

- The literature is dominated by RP and PS. Other algorithms may be good on average but fail only occasionally to satisfy various axioms.

Our idea

- The literature is dominated by RP and PS. Other algorithms may be good on average but fail only occasionally to satisfy various axioms.
- For artificial agents, axioms involving envy are often not relevant; strategy and efficiency are also less important. Perhaps overall welfare is a better design criterion.

Our idea

- The literature is dominated by RP and PS. Other algorithms may be good on average but fail only occasionally to satisfy various axioms.
- For artificial agents, axioms involving envy are often not relevant; strategy and efficiency are also less important. Perhaps overall welfare is a better design criterion.
- We perform exhaustive computation for small $n = m$ and measure welfare, violations of envy-freeness, violations of SD-efficiency and violations of SD-proportionality. We also do some simulation for larger n.

Our idea

- The literature is dominated by RP and PS. Other algorithms may be good on average but fail only occasionally to satisfy various axioms.
- For artificial agents, axioms involving envy are often not relevant; strategy and efficiency are also less important. Perhaps overall welfare is a better design criterion.
- We perform exhaustive computation for small $n = m$ and measure welfare, violations of envy-freeness, violations of SD-efficiency and violations of SD-proportionality. We also do some simulation for larger n.
- Overall results show that (if we run TTC on the output of YS and NB) that YS gives clearly better welfare than the other algorithms, with increased probability of violating envy-freeness and efficiency.

Summary of results

- PS is best for envy-freeness and efficiency.

Summary of results

- PS is best for envy-freeness and efficiency.
- YS+TTC is best for utilitarian and Nash welfare when Borda utilities are used.

Summary of results

- PS is best for envy-freeness and efficiency.
- YS+TTC is best for utilitarian and Nash welfare when Borda utilities are used.
- NB is best for welfare when "plurality utilities" are used.

Sample results: efficiency and welfare

Algo	Util	Egal	Nash	Eff
PS	0.950	3.909	4.256	1
RP	0.946	3.818	4.234	0.428
NB	0.958	3.400	4.247	0.892
YS+TTC	0.980	3.833	4.380	0.906

Table: IANC5

Sample results: fairness

Algo	%EF	#EF	w%EF	w#EF	%Prop	#Prop	w%Pr
PS	1	1	1	1	1	1	1
RP	0.0919	0.840	1	1	1	1	1
NB	0.0774	0.828	0.731	0.982	0.253	0.778	1
YS+TTC	0.0796	0.818	0.620	0.975	0.288	0.807	0.998

Table: IANC5fair

Algo	size 10	15	20	25	30	35	40	45
PS	0.952	0.959	0.964	0.968	0.972	0.974	0.977	0.
RP	0.937	0.941	0.946	0.950	0.953	0.956	0.959	0.
NB	0.949	0.951	0.954	0.957	0.960	0.963	0.965	0.
YS+TTC	0.977	0.979	0.981	0.983	0.984	0.986	0.987	0.

Table: Borda utilitarian efficiency for larger n under IC

Future work

- Use preference distributions with more correlation between agents: in particular the IC (IID uniform) distribution for large n yields few conflicts, allowing each agent to get one of her top few choices with high probability.

Future work

- Use preference distributions with more correlation between agents: in particular the IC (IID uniform) distribution for large n yields few conflicts, allowing each agent to get one of her top few choices with high probability.
- Analytic results under IC distribution?

Future work

- Use preference distributions with more correlation between agents: in particular the IC (IID uniform) distribution for large n yields few conflicts, allowing each agent to get one of her top few choices with high probability.
- Analytic results under IC distribution?
- Investigate Yankee Swap in more detail. Is it somehow related to the Gale-Shapley algorithm for two-sided matching? Could it be useful for school choice?